2010

The ATSA method for specifying both system and user interface requirements: an application of activity theory

Robert Bruce Keanan Brown

University of Wollongong

Recommended Citation
NOTE

This online version of the thesis may have different page formatting and pagination from the paper copy held in the University of Wollongong Library.

UNIVERSITY OF WOLLONGONG

COPYRIGHT WARNING

You may print or download ONE copy of this document for the purpose of your own research or study. The University does not authorise you to copy, communicate or otherwise make available electronically to any other person any copyright material contained on this site. You are reminded of the following:

Copyright owners are entitled to take legal action against persons who infringe their copyright. A reproduction of material that is protected by copyright may be a copyright infringement. A court may impose penalties and award damages in relation to offences and infringements relating to copyright material. Higher penalties may apply, and higher damages may be awarded, for offences and infringements involving the conversion of material into digital or electronic form.
The ATSA Method for Specifying both System and User Interface Requirements:

An Application of Activity Theory

A thesis submitted in fulfilment of the requirements for the award of the degree

DOCTOR OF PHILOSOPHY

From

THE UNIVERSITY OF WOLLONGONG

by

ROBERT BRUCE KEANAN BROWN

B.Arts (Honours Class II.1) UOW

School of Computer Science and Software Engineering

2010
Acknowledgements

Hier stehe ich. Ich kann nicht anders.
Gott helfe mir. Amen

Here I stand. I can do no other.
God help me. Amen

- Martin Luther (Apocryphal)

Spoken in self defence against charges of heresy
before the Diet of Worms, April 17, 1521

I would like to thank so many people for their patience, support, wisdom and input that I fear the available space in this acknowledgement section precludes full and precise enumeration. I must though, in particular, find the space to offer profound thanks to my Supervisors, Dr Ian C Piper and Associate Professor Peter Hyland, without whose extraordinary patience and stamina, inspiration, advice and long-suffering tolerance, no single word of this thesis could have ever seen the light of day.

As for the rest of you, allow me to respond with the words of my 1980’s High School Principal, “you know who you are, don’t make me come and get you!”

Luther before the Diet of Worms
Photogravure after the historicist painting by Anton von Werner (1843-1915) in the Staatsgalerie Stuttgart

Abstract

The many diverse practices of computerised information systems analysis and design, both of the systems themselves and their human computer interfaces, has only further mired themselves more deeply with each passing methodology, method, technique or tool; including those which reject Methodism and strive for a near Zen state of ‘structurelessness’.

The stakeholders and customers whose personal and business lives rely so deeply on the facilitating tools crafted by the artisans of these ‘design’ practices deserve better than the disappointingly low success rates delivered to date. No amount of obfuscation or rhetoric can hide the embarrassing reality that these professions cannot reliably and predictably perform their jobs to an acceptable and reliable level. Few other practices aspiring to be professions could tolerate such a history, and it invites speculation as to the veracity of a claim to professional status.

A review of the literature reveals challenges for the design artisan and their methods: they lack a single coherent informing philosophical or theoretical base; they lack clear lines of communication between themselves, their customers and the stages of their work; they rely too heavily on accumulated experiences, difficult to transfer to neophytes whose entry to professional practice was discouraged. Finally, customer requirement capture was poorly understood and poorly conducted.

There are grounds to consider selecting an informing theory which readily grasps the complexities and scaling issues of user organisations. It was decided that Activity Theory showed promise, despite no demonstrably complete systems analysis and design method (inclusive of interface) based on Activity Theory having been located in the literature.

The hypothesis was formed, that:

Systems Analysis and Design (SAD) may be improved if conducted through a prescriptive but agnostic method, constructed according to Activity Theory (AT) principles.

The thesis describes in narrative detail how just such a method was constructed and tested for indications of feasibility under a Normative Research Methodology and an Ideational Evaluation.

It is hoped that the method produced can provide a consistent, learnable and lightweight framework for future practice; if only for early career neophytes, such that they may work with each other and their clients to produce workable and acceptable results, whilst they accumulate the tacit experience necessary to allow for more elegant future designs.
Thesis Certification

CERTIFICATION

I, Robert Bruce Keanan Brown, declare that this thesis, submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Computer Science and Software Engineering, Faculty of Informatics, University of Wollongong, is wholly my own work unless otherwise referenced or acknowledged. The document has not been submitted for qualifications at any other academic institution.

Robert Bruce Keanan BROWN

16th November, 2010
# Table of Contents

Acknowledgements........................................................................................................................................... i  
Abstract ............................................................................................................................................................ iii  
Thesis Certification ........................................................................................................................................ v  
Table of Contents ........................................................................................................................................ v i i  
List of Tables ................................................................................................................................................ x i i i  
List of Figures ............................................................................................................................................... x v  
List of Publications ......................................................................................................................................... x i x  

## Chapter 1 Introduction ................................................................................................................................ 1  
1.1 Background and Motivation ..................................................................................................................... 1  
1.2 Research Goals and Scope ..................................................................................................................... 2  
1.3 Significance of the Thesis ....................................................................................................................... 4  
1.4 Structure of the Thesis .......................................................................................................................... 5  

## Chapter 2 Literature Review ..................................................................................................................... 9  
2.1 Introduction ............................................................................................................................................. 9  
2.2 The Perceived Low Success Rate of Software Projects ......................................................................... 9  
  2.2.1 Standish Group’s Infamous CHAOS Reports .................................................................................... 10  
  2.2.2 More Recent Reflections on Failure Rates ..................................................................................... 11  
2.3 The Software Crisis and Some Structure ............................................................................................. 12  
  2.3.1 Boehm’s Hegelian History of SE ................................................................................................... 13  
  2.3.2 NCC and the Traditional BCS Method ............................................................................................ 13  
  2.3.3 The Structured Systems Analysis Approach .................................................................................. 13  
  2.3.4 The Software Development Lifecycle (SDLC) .............................................................................. 15  
  2.3.5 Evolutionary Development ............................................................................................................ 16  
  2.3.6 Incremental Delivery ....................................................................................................................... 16  
  2.3.7 The Spiral ....................................................................................................................................... 16  
  2.3.8 Object-Orientation (O-O) .............................................................................................................. 18  
  2.3.9 Agile (Lightweight) Methods ........................................................................................................ 25  
2.4 Sidebands of the SDLC .......................................................................................................................... 29  
  2.4.1 Business Process Mapping/Modelling (BPM) ............................................................................... 29  
  2.4.2 The Human Computer Interface (HCI) ......................................................................................... 34  
2.5 Reflections Thus Far .............................................................................................................................. 40  
  2.5.1 Of Methods and Methodologies .................................................................................................... 40  
  2.5.2 A Software Engineering Crisis? - Method Fatigue? .................................................................... 41  
  2.5.3 Is It All About Requirements? ...................................................................................................... 44  
  2.5.4 Turning Requirements into Specifications ...................................................................................... 45  
2.6 The Need for a Theory ......................................................................................................................... 46  
  2.6.1 Towards Activity Theory .............................................................................................................. 46  
  2.6.2 First Generation AT ...................................................................................................................... 48  
  2.6.3 Second Generation AT .................................................................................................................. 48  
  2.6.4 Third Generation AT ..................................................................................................................... 49  
  2.6.5 AT’s Historical Dimension ............................................................................................................ 49  
  2.6.6 Engström’s Four AT Contradictions .............................................................................................. 50  
  2.6.7 More Recently… .............................................................................................................................. 51  

Chapter 4

4.11 Components of the AT Model Based Method ............................................................... 104
4.11.1 The Five Ss ................................................................................................................. 105
4.11.2 The Four Scratchboards ........................................................................................... 107
4.11.3 Grey Box Specification .............................................................................................. 107

Chapter 3

3.4.3 Two Dimensional Research Framework ........................................................................ 78
3.4.2 Design Science Artefacts .............................................................................................. 76
3.4.1 Theoretical Background ............................................................................................... 75
3.3 Choosing a Research Framework ........................................................................................ 74
3.2 Research Goals .................................................................................................................. 74
3.1 Introduction ........................................................................................................................ 73

Chapter 2

2.8 Putting AT to Work ............................................................................................................. 56
2.8.1 Bødker ............................................................................................................................. 56
2.8.2 Kuuti ................................................................................................................................ 57
2.8.3 Fjuk et al. .......................................................................................................................... 57
2.8.4 Kaptelinnin et al. ............................................................................................................. 57
2.8.5 Gifford and Enyedy ........................................................................................................ 58
2.8.6 Jonassen and Rohrer-Murphy ....................................................................................... 58
2.8.7 Korpela .......................................................................................................................... 59
2.8.8 Mwanza ........................................................................................................................... 60
2.8.9 Fjeld et al. ....................................................................................................................... 62
2.8.10 Collins et al. .................................................................................................................. 62
2.8.11 Vrazalic .......................................................................................................................... 63
2.8.12 Martins .......................................................................................................................... 65
2.8.13 Uden et al. ..................................................................................................................... 65
2.8.14 And Beyond ................................................................................................................... 67

2.9 Findings: Challenges for SAD and HCI ............................................................................. 69
<table>
<thead>
<tr>
<th>Appendix</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Glossary of Acronyms</td>
<td>281</td>
</tr>
<tr>
<td>B</td>
<td>Listing Business Company Procedures</td>
<td>283</td>
</tr>
<tr>
<td>C</td>
<td>Business Resources</td>
<td>299</td>
</tr>
<tr>
<td>D</td>
<td>Original Business Process Diagrams</td>
<td>301</td>
</tr>
<tr>
<td>E</td>
<td>List of Goal Directed Actions</td>
<td>303</td>
</tr>
<tr>
<td>F</td>
<td>List of Single Instance Nodes (SINs)</td>
<td>309</td>
</tr>
<tr>
<td>G</td>
<td>Activity Conflation Worksheet</td>
<td>325</td>
</tr>
<tr>
<td>H</td>
<td>Activity Shells</td>
<td>339</td>
</tr>
<tr>
<td>I</td>
<td>Combined Activity Table (CAT)</td>
<td>361</td>
</tr>
<tr>
<td>J</td>
<td>Node Reduction Heuristic</td>
<td>363</td>
</tr>
</tbody>
</table>
List of Tables

Table 2.1: Usability Heuristics (from Nielsen (1994)) ................................................................. 35
Table 3.1: Two-dimensional research framework (from March and Smith (1995)) ......................... 78
Table 3.2: Criteria to evaluate artifacts (from March and Smith (1995)) ........................................ 79
Table 4.1: An Extended AT Taxonomy .......................................................................................... 104
Table 4.2: Vrazalic’s DUEN Elicitation Questions - with commentary ............................................. 117
Table 4.3: Proposed Phase One Elicitation Questions ...................................................................... 118
Table 4.4: Some Candidate SFA Command Terms .......................................................................... 133
Table 4.5: Initial Doings, Candidate Activities ............................................................................... 145
Table 4.6: The CAT Adjacency Matrix for the Activity Network in Figure 4.25 ............................... 147
Table 4.7: Instrument Table, listed by Activity .............................................................................. 154
Table 4.8: Revised AT Taxonomy .................................................................................................... 159
Table 4.9: Questions to Assist in Eliciting Transactions .................................................................. 165
Table 4.10: Method Terms Before and after AT Analysis and Redesign .......................................... 180
Table 5.1: The ATSA Method, Stepwise ......................................................................................... 191
Table 5.2: Positions ......................................................................................................................... 193
Table 5.3: Roles ............................................................................................................................... 197
Table 5.4: Business Procedure Selection ......................................................................................... 198
Table 5.5: Identified Business Documents ...................................................................................... 202
Table 5.6: Revised AT Taxonomy ..................................................................................................... 203
Table 5.7: Model Activity Shell ....................................................................................................... 208
Table 5.8: Meanings of Method Terms after Redesign ................................................................. 210
Table 5.9: Field Descriptors used in the Instrument List ................................................................. 215
Table 5.10: Some Sample Instrument Records .......................................................................... 217
Table 5.11 Roles currently assigned to the Office Manager .......................................................... 219
Table 5.12: Activity Shells for JobA ................................................................................................. 222
Table 5.13: JobA Instruments ........................................................................................................ 222
Table 5.14: Activity Shells for EmpA .............................................................................................. 232
Table 5.15: EmpA Instruments ....................................................................................................... 232
Table 6.1: Criteria to evaluate artifacts (from March and Smith (1995)) ............................................ 237
Table 6.2: Activity Shell 26, from Appendix G ................................................................................... 245
Table 6.3: Some Sequenced Transactions from Activity 26 ................................................................. 246
List of Figures

Figure 2.1: An Hegelian History (from Boehm (2006)) ................................................................. 14
Figure 2.2: The Waterfall Lifecycle (from Satzinger et al. (2009)) .................................................. 15
Figure 2.3: The Spiral Lifecycle (from Boehm (1988)) ................................................................. 17
Figure 2.4: A History of O-O (from Vinci (2004)) ......................................................................... 21
Figure 2.5: A Generic Use Case (from Booch et al. (2000)) .......................................................... 22
Figure 2.6: Multiple Use Cases, overlayed to investigate the system .............................................. 23
Figure 2.7: A Complex of Use Cases, inverted to investigate the network of users ....................... 23
Figure 2.8: Evolutionary Map of Agile Methods (from Abrahamsson et al. (2003)) ....................... 26
Figure 2.9: Detail from Boehm’s (2006) Hegelian History [emphasis added] ............................... 27
Figure 2.10 An SD Model (from Yu (1999)) ................................................................................. 31
Figure 2.11: An SR Model (from Yu (1999)) .................................................................................. 31
Figure 2.12: Waterfall and classic HCI iterative design (from Lauesen (2005)) ............................. 39
Figure 2.13: AT conception (from Vygotsky (1978)) ..................................................................... 48
Figure 2.14: AT hierarchy (from Leont’ev (1978)) ....................................................................... 48
Figure 2.15: AT matrix (Engström (1987)) .................................................................................... 49
Figure 2.16: AT Sub-Triangles (from Mwanza (2001)) ................................................................. 61
Figure 3.1: Constructs, models, methods and instantiations (from March and Smith (1995)) ........ 78
Figure 4.1: Divergence of implementation from elicited direction ............................................... 93
Figure 4.2: A possible change of emphasis between stages of a product lifecycle ....................... 95
Figure 4.3: An Early Workflow Concept ......................................................................................... 96
Figure 4.4: An Early Conception of an Activity Theoretic Analysis ............................................. 99
Figure 4.5: Second Workflow Concept ......................................................................................... 100
Figure 4.6: Mapping Activity Theoretic Elements to UI Components ............................................ 105
Figure 4.7: The 5S Concept - the ‘Kidney Diagram’ ....................................................................... 106
Figure 4.8: Third Workflow Concept ............................................................................................ 110
Figure 4.9: Third Workflow Concept – Lecture Version ............................................................... 111
Figure 4.10: Fourth Workflow Concept - The ‘U Diagram’ ............................................................ 112
Figure 4.11: Basic Entity Relationship Structure ......................................................................... 122
Figure 4.12: Sample Activity Record Form ................................................................. 123
Figure 4.13: Sample Action Record Form ................................................................. 123
Figure 4.14: Sample Operation Record Form ............................................................ 123
Figure 4.15: Sample ‘People’ Record Form ............................................................... 124
Figure 4.16: Sample Role Record Form ................................................................. 124
Figure 4.17: Basic Entity Relationship Structure with Roles ................................. 126
Figure 4.18: Augmented ‘U’ Diagram – The ‘3T Diagram’ ...................................... 129
Figure 4.19: Sample ‘Constraint’ Record Form ....................................................... 134
Figure 4.20: Sample Instrument Record Form ....................................................... 138
Figure 4.21: Basic Entity Relationship Structure with Instruments ......................... 139
Figure 4.22: Tool nodes with Instruments tangential to the System Boundary ......... 140
Figure 4.23: The ‘Amended U diagram’ .................................................................. 143
Figure 4.24: Fifth Workflow Concept – the ‘W diagram’ ........................................ 144
Figure 4.25: Reference Model of the Activity Network ............................................ 146
Figure 4.26: Activity Network with Pipes and Joints Highlighted ......................... 148
Figure 4.27: Partially Refined Activity Network .................................................... 149
Figure 4.28: Partially Refined Activity Network with SDR .................................... 151
Figure 4.29: Initially identified system Activities, Instruments and the SDR .......... 152
Figure 4.30: Duties to Doings. Roles after Redesign ............................................. 156
Figure 4.31: Some Entity Relationships from the Trial Case ................................. 160
Figure 4.32: Decomposing SINs .............................................................................. 168
Figure 4.33: Conflating Activities ......................................................................... 169
Figure 4.34: Overview of the ATSA Method ......................................................... 179
Figure 5.1: AT matrix (from Engström (1987)) ..................................................... 208
Figure 5.2: Job Sheet for sample job 012345-123 ............................................... 224
Figure 5.3: Job Map for sample job 0012345-123 ............................................... 225
Figure 5.4: Client Contact Details for sample client ........................................... 225
Figure 5.5: Sample pop-up Job Scope Questions list .......................................... 226
Figure 5.6: Confirmation Dialogues to prevent illegal assignments .................... 233
List of Publications

Refereed Journal Publications


Refereed Conference Publications


Chapter 1  Introduction

To orient the reader, this chapter provides, with broad brush strokes, a summation of the motivation and background of study. The research goals and scope of the study are then outlined. The significant features of the research are outlined, highlighting its novel and positive contribution. Finally, an outline of the thesis, chapter by chapter is presented.

The reader is presented less with an exploration of the natural world and quantitative test of claims about it than with an investigation of the constructed world of human artefacts and a normative ideational test as to the feasibility of constructing a novel artefact. The artefact was designed specifically to see if specific principles of its construction, in this case, the Activity Theory (AT, see section 2.6), which it is hypothesised may assist with some extant challenges, could produce just such an artefact. In proposing an entirely new Systems Analysis and Design (SAD, see sections 2.3-2.5) method (the artefact), it is claimed in reference to the Normative Research methodology (Simon 1969, March and Smith 1995, Routio 2005) and ideational evaluation techniques (Iivari 1991), that it suffices to call for its initial construction in accord with particular notions, to details that construction, and to demonstrate its feasibility.

The corollary should follow, that if the novel artefact proved feasible, future work extending the capabilities of the artefact might permit more quantitative and comparative evaluations of its function against that of mature pre-existing artefacts built under previous informing principles. Such follow up work, if added to the initial investigation, construction and feasibility test presented here would greatly exceed the scope of any normal doctoral thesis and so are not included. It falls to those who wish to extend upon the work presented in this thesis to conduct such quantitative studies.

1.1  Background and Motivation

This study followed a somewhat unorthodox route, bringing into fields more accustomed to quantitative validation and the metrics of comparison, a sometime narrative historical argument built under the precepts of normative research and ideational evaluation.

This thesis sits astride challenging disciplinary boundaries. It might be argued that whilst systems analysis and interface design are commonly considered separate activities they share sufficient proximity to be discussed together. To bring the issue of the design of system interfaces and broader issues of general software engineering methods and approaches together invites dispute. In its earliest published expressions, this body of notions was often queried by audiences who were almost always of one camp or the other. One niche workshop, in Rome in 2005 (Integrating Software Engineering and Usability Engineering, Workshop 11 of INTERACT (2005) Rome), actively sought to invite cross-fertilisation between requirements engineering and usability engineering, and even there, the novelty of suggesting a workable SAD application for AT raised eyebrows.

The author had a history of study in the classical civil engineering domain, in the history and philosophy of science and technology, considerable experience as a
vocational trainer and extensive experience as a graphical designer of industrial and commercial products, including interface elements for web applications and other software. This curious mixture motivated an interest in abstracted notions of prescriptive ‘engineering-flavoured’ methodologies and their informing philosophies, the manner in which they are taught, and their application to user interfaces. The author suffered from a profound desire to be able to describe in some useful way, what was embodied in the very concept of “designing”.

Studying the construction and application of information systems and the technologies which support them, quickly highlighted concerns as to the principles underlying choices in their designs. Several years of designing and delivering undergraduate and post-graduate academic courses on these topics such as Human Computer Interaction (HCI), likewise, highlighted concerns as to the manner in which new entrants to the discipline are taught, and the suitability of methods for neophyte ‘designers’.

Vrazalic’s 2004 PhD thesis (2004) introduced an intriguing application of the somewhat obscure, oft cited but rarely understood AT. It promised, perhaps for the first time, a prescriptive method by which part of the lifecycle of a software project (in this case, post deployment usability testing and evaluation) could be conducted under the precepts of AT. The author’s initial reaction was to question the efficacy of work-intensive preparation of test cases after project delivery; could it be possible to transfer that work to the design stage and use it to inform construction (which would hopefully then be more likely to conform as required) as well as preparing post-project test cases. A secondary reaction was to resolve the significant cognitive load on the analyst-designer imposed by AT by assembling a readily teachable prescriptive method. Perhaps such an ‘early stage’ application of AT might have currency throughout much of the lifecycle. A final reaction was to consider whether AT could deliver some détente between the seemingly incommensurable worlds of the system designer and the interface designer.

Armed with these musings, a literature review supported a number of these concerns, and highlighted a small but growing number of calls for the application of AT. The arduous and difficult task of assembling a novel Activity Theoretic method from first principles, deliberately agnostic as to extant approaches and as independent as possible of final technological choices for implementation had begun.

1.2 Research Goals and Scope

Although the structure of the research goals is more clearly explained in chapter 3, for the sake of the readers’ convenience, the stated research goals pursued in this thesis are foreshadowed here:

1. To identify challenges in existing SAD methods.
2. To hypothesise some means of improvement to address the challenges.
3. To construct a SAD method, based on the hypothesis.
4. To evaluate the new method against the identified challenges, by applying it against a realistic case study.
The clear statement of research goals emerged, as it must, from a directed survey of the literature which is presented in the next chapter. Some challenges in the broad field of SAD are identified and further investigation leads to the hypothesis that the use of an overall informing theoretical basis may assist in addressing some of these challenges. Some prior literature suggests AT as a strong candidate in that role.

Whilst this thesis followed a somewhat traditional arrangement of chapters, presenting units essentially analogous to: introduction, literature review, experimental design, experimental test, discussion of results etc; the function here was not identical to a classical science experiment. Under classical science conditions an hypothesis resulting from the literature review would result in one or more testable claims. An experimental apparatus would be constructed to test those claims and then the claims would be submitted for testing to produce some conclusion as to the validity of the hypothesis. In this instance however, the hypothesis produced a claim that if an artefact (in this case, a novel activity theoretic SAD method) could be produced, it may be of some use in addressing challenges to extant methods which may lack some of the benefits imbued by the special characteristics of the novel methods construction in a certain way (in this case, under the principles of AT). In a manner of speaking, it might be said that the “experiment” presented here tests if the experimental apparatus could be built at all. The research methodological framework for such styles of investigation was laid out in detail in chapter 3.

As mentioned above, the field of usability and HCI were previously familiar to the author and served initially as a conveniently more containable microcosm in which to explore challenges found across the lifecycle of software projects (the analysis of requirements, design and implementation of a User Interface (UI) follows essentially the same core processes and is a necessary element of larger software projects in any event). HCI and usability is also the field where AT seems to have had its first airings to an information technologic audience. Furthermore, the doctoral thesis work of Vrazalic (2004) strongly informed early research motivation in this project, and was tightly bound to the evaluation of systems’ interface usability. HCI seemed a natural sandbox in which to initially conduct this research.

Though the challenges for SAD existed across the domain, it was felt initially that a realistic scope for constructing and evaluating a novel activity theoretic analysis and design method might contain it to the realm of HCI and UI. As indicated in chapter 4 however, the novel method built to address the research goals quickly showed the promise of applicability in the broader field of systems analysis and design. The final method, as described at the end of chapter 4 and evaluated ideationally in chapter 5, focuses primarily on the analysis and design of whole systems and the notion of specifying interfaces at that time was limited to the description of functions and how they would best be clustered into screens, sets-of-screen and individual end-user workstations. Investigation as to the novel methods applicability for more minutely detailed interface specification (though initially considered in early stages of chapter 4) was ultimately deemed beyond the reasonable scope of the project, and assigned for possible investigation in future work. A claim for the novel method however was that it assembled a requirements elicitation and design specification method which
more closely aligned systems and their interfaces than was typical for extant methods.

Whilst the novel system strove to remain agnostic of extant methods and design philosophies, and non-aligned with any specific code languages or implementation technologies and methods, for the purposes of the ideational evaluation in chapter 5 it proved necessary to demonstrate that the method could do what extant SAD methods can do. To that end, after an elicitation and analysis phase, the final ‘redesign’ phase of the novel method produced a specification after the manner of a ‘data dictionary’ and ‘process list’ which might be identified with the traditional structured methods of systems analysis and design. This scoping decision was taken specifically for ease of demonstration, as the traditional structured approaches are, at least, known and recognisable, even by their more strident opponents.

The claim is made that the novel method built in this research project is not unfriendly towards other approaches (such as Object Orientation or Agile methods) and could readily be adapted to suit them. AT informs an analysis which drives models from which Process can be read (as was demonstrated) or from which structured data ‘Object-like’ entities (and their methods and use-cases) might be defined. If the models of the method can be shown to be buildable in a quick and lightweight enough manner, then it may even serve as an entry for some of the later forms of Agile approaches, such as Agile Modelling (AM). It remains the focus of possibly several future research projects for these articulations to be realised in prescriptive form.

It was also deemed beyond the scope of the research project to construct semi-automated analysis and design tools, in the manner of Computer Aided System Engineering (CASE) tools. This too was set aside as a possible future research project.

The ideational evaluation conducted in chapter 5 was limited to the analysis and design of a relatively typical real-world small business. The business was highly interactive, with the principle stakeholders working in close proximity. The design of a facilitating system, by way of the novel method, was deliberately gentle with and sensitive to the pre-existing business processes. This was not to suggest that the method could frame more radical process surgery, but rather, to demonstrate that extant “work styles” and “business cultures” could be preserved under an activity theoretic SAD method.

1.3 Significance of the Thesis
This research study has produced a SAD method which is based on AT. Construction of this novel method aimed to address challenges associated with current system, UI and HCI techniques of analysis and design.

According to the principles of ideational evaluation (Iivari 1991), as the novel method was demonstrated to be able to do what other SAD methods can do (and perhaps more so, in some regards), then since it had been constructed in accordance with the hypothesis which addressed the challenges identified in other SAD methods, the thesis was established.
By establishing that an activity theoretic SAD method is feasible, this thesis offers the promise of some reconciliation between the currently segregated worlds of SAD and HCI.

By demonstrating that a single, consistent informing theoretical base can underpin a method, this thesis offers a way forward to reconciling the breakdown of current methods into ad hoc hybrids (as happens in practice).

This thesis further offers a framework both deliberately agnostic of pre-existing methods and not unfriendly to any major implementation technology. It could therefore serve as a unifying ‘spine’ onto which the practitioner could still attach many of the current techniques, thus imbuing them with the intellectual framework needed to link them to the other elements of the software projects lifespan, a ‘methodological middleware’.

Finally, this thesis offers the hope that, under a single theory, neophyte practitioners might be able to learn their trade more readily, and be able to produce at least feasible analyses and designs (consistent with interface specifications) in their early career. As experience builds, their work would be expected to gain elegance and efficiency, but the barrier of the “appeals to prior experience” which seems to be needed to use many current methods may weaken as a result.

If the method demonstrated in this thesis, still in its infancy, were only to find a niche as a ‘teaching method’, then even that would be no small contribution to the field.

1.4 Structure of the Thesis

This thesis consists of seven chapters. Chapter 1 has broadly described the research study concentrating on the background and motivation, the research goals and scope and stated the significance of the thesis.

Chapter 2 provided a review of literature across the fields concerned, none of which are clearly or consensually agreed. Such a broad area of concern necessitated a somewhat wider-than-deeper study, sufficient to identify areas of concern and explore possible responses which could lead to the formulation of a testable hypothesis.

Chapter 3 details the choice and deployment of the research methodology adopted for this study. The purpose of this research was not to test or compare methods (whose prior existence is clearly required), but to construct an entirely new novel method. The product of this act of creation was the result. A classical quantitative research methodology would therefore be inappropriate. To both build from first principles, and then exhaustively test, a novel (ie: not yet mature or widely deployed) method would vastly exceed, by a factor of at least two, the scope for any reasonable doctoral thesis.

Simon (1969) described research which produces artefacts to service human requirements under the title Design Science (distinct from natural science) and a research methodology was chosen consistent with this notion. This was research concerned with ‘production’ not with ‘discovery’, ‘comparison’ or ‘measurement’.
Chapter 1 - Introduction

Following the framework laid down by March and Smith (1995), a Normative Research Methodology together with ideational evaluation were described and adopted for the study. The adopted research methodology framework itself provides structure to the research goals briefly mentioned above. The four steps adopted for the research methodology, according to the work of Routio (2005) were:

1. evaluative description of the initial state, defining the need for improvement
2. analysis of relationships and possibilities for change
3. synthesis of a proposed improvement
4. evaluation of the proposal

Chapter 4 is a comprehensive narrative account of the construction of the novel activity theoretic method. This was assembled in accordance with the second stage of the chosen research methodology.

The reader is presented with a full account of the thought processes which were followed, even those which would prove to be false starts and blind alleys. The motivation is to present the reader with a clear understanding of how the novel method came to be such that there can be no doubt as to its fidelity and truthfulness to concept. It will be seen that, as might be expected in a domain rife with dozens of methods and techniques, many components of extant methods bare an uncanny resemblance to the method constructed. It was felt that a bland statement of the final product could appear to be a simple pastiche of extant component notions.

The core principle informing the method constructed was its consistent and at times slavish adherence to the precepts of a single, consistent informing theory (namely, AT). It proved necessary to narrate construction historically, to clearly demonstrate otherwise. The chief product of this thesis is the method constructed in chapter 4, and if only for that reason, its creation warranted a detailed examination.

Chapter 5 presents an evaluative case study in which the novel method does, indeed, perform substantially the same functions as extant SAD methods. This was performed in accordance with the final step of the research methodology, under the precepts of Ivari’s (1991) ideational evaluation framework.

Chapter 6 discusses the results of the ideational evaluation conducted in chapter 5. The extent to which the novel method addressed the challenges identified in chapter 2 for SAD methods were discussed, as were the claims for the novel method.

Chapter 7 makes concluding remarks, and drew together the findings of the study and contributions of the research, identifying the limitations of the study. Suggestions for a number of future research projects were given, including preliminary details for a follow-up qualitative study which was already underway at the time of this writing.

After a bibliographic listing of academic references used throughout this thesis, details of the evaluative case study from chapter 5 are presented in a number of appendices, referred to largely through chapter 5 as they contain unwieldy tables of case study results which would have impeded the readability of the thesis proper. A CD-ROM included with the printed and bound copies of this thesis contains a
spreadsheet file that allows for a better presentation of case study details, including some which were not suitable to presentation in hardcopy format.

Having outlined the motivations, origins, goals and significance of this research study and the structure of its chapters, it is necessary to review domain literature to establish challenges facing extant SAD methods, including the design of their user interfaces.
Chapter 1 - Introduction
Chapter 2  Literature Review

2.1  Introduction
The background and motivations outlined in chapter 1 highlighted a number of concerns with the concepts of analysis and design in information systems and their facilitating technologies. The Vrazalic thesis (2004) provided a starting point for considerations that perhaps some application of Activity Theory (AT), if feasible, may be of potential benefit.

It became necessary to survey the broad array of literature in the field to establish the concerns and challenges and to validate the notion of choosing AT.

As there was no consensus of the field of systems analysis and design, to the point that, from one perspective all notions could exist broadly under the topic ‘software engineering issues’ whilst from another ‘software engineering’ was but one in an historic parade of themes; any such survey of literature would have to have been broad. Extensive investigation under any single area would have invited a paradigmatic skew which, if allowed its head, would colour the readings of all the other points of view.

To give full justice to the wealth of literature would require such an exhaustive review as to fill an entire doctoral thesis several times over. The motivation for this current research project was most emphatically not to simply generate an index, outline or lexicon for a complex multi-faceted field (however much there was a need for just such a thing), but rather to attempt some meaningful contribution. This goal necessitated that the literature review should start with the reasons a contribution might be necessary; the notion of poor success rates to date.

2.2  The Perceived Low Success Rate of Software Projects
In civil engineering, failure can readily lead to injury and death as well as massive financial losses. The iCivilEngineering website (www.icivilengineer.com) maintains a watch list of civil engineering project failures from around the world. Whilst it is far from exhaustive, and makes no claim to be, at the time of this writing it listed around 40 failure incidents (each resulting in injury or death) from around the world, between May 2000 and July 2006. Each of these events is regrettable in the extreme, but even if it was assumed that only 10% of failures were reported on the site, the failure rate would barely approach a few percent of the total number of civil engineering projects.

Whilst loss of life due to IT project failures is considered rarer (exceptions being cases such as the infamous Therac-25 failure (Leveson & Turner 1993) the ‘elephant in the room’ whenever information systems or software project developments were under discussion has been the broadly acknowledged, but politely obscured, massive failure rate. It has been this author’s personal experience that to raise the topic in ‘polite’ (professional) society is to invite silence, ostracism or even an abrupt retort, perhaps best summarised as “yes, yes we know, stop going on about it!” Citation of infamously high failure percentage figures from the 1990’s (detailed below) are often dismissed as “outdated” or “no longer valid”. To date, no statistics have been
published citing success percentages that radically differ from those which are not-to-be-mentioned.

Not infrequently, motivations behind and justifications of new approaches, methodologies, methods, techniques and the like are commonly framed as mild to vociferous attacks on one or more predecessors’ shortfalls, rather than as a direct attempt to address the massive and embarrassing failure rate.

As a starting point for this review of literature, it was necessary to present what evidence could be found to believe something is seriously awry with the way our software projects are being realised.

2.2.1 Standish Group’s Infamous CHAOS Reports

Several frequently cited surveys of the late 1990’s suggest that there is a poor success rate in software projects. The CHAOS report (Standish 1995) suggests that over 30% of projects are not completed at all, that over 50% of projects run into significant cost overruns and that as few as 17% of projects are completed both on time and budget.

The CHAOS’95 report elicited opinions as to the high rate of software project failure. It reported the following breakdown of responses across 11 contributing factors:

- Incomplete requirements 31.1%
- Lack of USER involvement 12.4%
- Lack of resources 10.6%
- Unrealistic expectations 9.9%
- Lack of executive support 9.3%
- Changing requirements & specifications 8.7%
- Lack of planning 8.1%
- Didn’t need it any longer 7.5%
- Lack of IT management 6.2%
- Technological illiteracy 4.3%
- Other 9.9%

It is important to note that poor requirements and a lack of user involvement rated as the highest perceived factors in the failure rate of software projects.

More recently, Crear (2009), CIO of the Standish Group, reports that the failure rates in 2009 “are low point in the last five study periods. [2009’s] results represent the highest failure rate in over a decade.” Crear’s comments highlight that, with some variations over time, failure rates remain unacceptable.

Even if software failure rates were being exaggerated by a factor of 10, they would be utterly unacceptable in any other field of engineering. If, for example, just seven in every hundred buildings, bridges or roadways failed, it would hardly be acceptable in any modern industrialised society of the 21st century.
2.2.2 More Recent Reflections on Failure Rates

The OASIG survey (1995) produced the famous ‘70% failure’ rate for IT projects. High abandonment rates and high cost overruns indicate probable misconceptualisation or misrepresentation of the purpose and requirements of the systems under development, requiring costly and time-consuming re-engineering effort. It would seem that the products under development are too often not the product that stakeholders and users actually want.

A widely cited 1997 KPMG Canada Survey of 1,450 key businesses (Whittaker 1999) reported a 61% IT project failure rate. The most commonly reported causes of failure were poor project planning, poor understanding of how the IT project should relate to the client organisation’s business needs, and a lack of top management support. Whittaker adds that another crucial factor was, “Poor estimates or weak definitions of requirements at the project planning stage.”

Wysocki and DeMichiell (1997) suggest failure rates of 70% or more and Hammer and Champy (1993) report a failure rate of 50%-70% for Business Process Re-Engineering (BPR) attempts.

The 1998 BULL survey conducted by Spikes Cavell & Co for Compagnie des Machines Bull (widely cited, including in Tichy and Bascom (2008)), conducted by Spikes Cavell in the UK to investigate major causes of IT project failure in the finance sector, found an inability to meet requirements accounted for 37% of failures, 75% missed deadlines and 55% exceeded their budget. More highly significant factors included communications breakdowns (indicative of a possible lack of a single informing methodology between stages of the lifecycle), a lack of planning and poor quality control.

Keil et al. (1998), citing figures from Johnson (1995) reported that of USD$250 billion spend on software projects in the USA through 1995, USD$59 billion was cost overruns and USD$81 billion was spent on failed projects. Perversely though, Linberg (1999) reports that “a team of software developers maintained a high-level of job satisfaction despite their failure to meet schedule and cost goals of the organization.” The notion that approaches taken by software developers to their work, at least in the late 90’s, (most probably inclusive of their various methods) were sadly lacking, is not easy to dismiss.

Lyytinen (1998) cautioned care with data of these kinds, especially when there could be mismatches of investigation method and problem, though he anticipated new classes of problem with information systems arising from an increasingly educated and sophisticated user base and new technologies. Far from playing the apologist, Lyytinen infers that there may have been even more issues with information systems that surveys and tests to that date had been crafted to detect.

The Robbins-Gioia 2001 survey (cited in Ellis 2007) reported that 51% of surveyed companies deemed their Enterprise Resource Planning (ERP) systems implementations to have been unsuccessful. The Conference Board survey (Cooke et al. 2001) found that 40% of ERP projects failed to meet their business case within a year of going live.
Goatham (2009) cited 2008 figures from the United States Government Accounting Office (GAO) that reported 49% of 840 federally funded were poorly planned, poorly performing or both. He backed this up with figures from the Information Systems Guild and Control Association, reporting 43% of 400 private industry respondents acknowledged a recent project failure. Goatham believed the apparent simplicity of projects (when represented in Gantt charts and the like) belied a bewildering complexity of decision points. For Goatham, these decisions were being made in an uncoordinated manner by an uncontrolled cluster of disconnected IT practitioners. Project decisions were being made by time-constrained IT professionals who had no governing body (unlike licensed construction engineers, for example), had as a result low entry level standards, and faced significant obstacles to the development of expertise. Goatham called these combined factors a "perfect storm for project failure.”

Looking forward 25 years, Gilb (2006) pessimistically anticipated ongoing “massive failure rates” for both small and large software projects, attributable largely to “bad requirements”, a lack of long-term thinking, a lack of architecture, poor software interfaces, an industry-wide obsession with “flavour of the month” buzzword methods, inattentive top management and ongoing delivery by Universities of “irrelevant” subjects.

2.3 The Software Crisis and Some Structure

Software engineering (SE) centres on two key concepts: ‘engineering discipline’ and ‘all aspects of software production’ (Sommerville 2001). Such ‘traditional’ software developments are product-centric. Some traditional software ‘myths’ disregard stakeholder objectives and the work then required after initial presentation to adjust the product to suit clients’ needs (Pressman 2001).

The SE community is recognising that inadequate requirements lead to increased likelihood of failure, especially on ‘softer’ socio-political grounds (Goguen 1993). This highlights the need to elicit stakeholder requirements and target their softer, qualitative objectives.

Maciaszek draws the production-centric distinction between ‘business rules’ – a functional requirement describing an ‘always on’ (invariant) aspect of the system, and ‘constraint statements’ which define restrictions on system behaviour or the production process (Maciaszek and Liong 2005). Whilst this distinction may not strictly address stakeholder preference, it allows acceptance or rejection on these grounds.

Stakeholders generate their own notations and terminologies, complicating the business of capturing such details (Sommerville et al. 1998). This difficulty informs approaches to requirement elicitation that are sensitive to consistency and viewpoint. It is necessary therefore to use a systematic approach when capturing requirements. A significant risk of failure exists in marginalizing the stakeholder’s softer objectives, despite their inherent messiness.

One approach to identifying stakeholder preferences adopts a goal-orientation and asks ‘what does the stakeholder want to achieve’. Goal formulations express
intended system properties (Lamsweerde 2001). Goal-centrism offers stability as top-level goals are often invariant under decomposition, and facilitate back-tracking when re-design issues arise.

Shifting to process orientation necessitates consideration of the software in its environment. It becomes necessary to identify active elements that have choice (Lamsweerde 2001). Such active elements (actors) are the loci for the formulation of goals and preferences. Process-centrism thus infers examination of actor goals, motivations (intentions), dependencies and cultural-historical activities.

2.3.1 Boehm’s Hegelian History of SE

Barry Boehm, who had been a significant player in the development of software from a very early stage, provided a useful tool when he assembled an Hegelian (that is, a deterministic plan of providence (Heidegger 1988)) roadmap of software engineering (2006) summarised by the chart given in Figure 2.1 (below). It is inserted in this review as it provides a convenient reference for locating notions, methods and approaches in a volatile domain.

2.3.2 NCC and the Traditional BCS Method

In the 1960’s the National Computing Centre of Great Britain (NCC) established a set of standards with techniques, documents and checklists Crinnion (1991). These incorporated a considerable degree of flexibility so that individual organisations could tailor them. They evolved into a six-week basic systems analysis course which, if passed, awarded a certificate of competency from the British Computer Society (BCS). These early methods were accompanied by tools such as the system flowchart, the procedure flowchart and the computer run chart. Crinnion, though an adherent in the early 1990’s of evolutionary design, attests to the efficacy of these early methods as systems built in the 1970’s were still in operation.

Crinnion observed that, despite the apparent solidity of systems built traditionally, they suffered from a use of inappropriate tools and a lack of user involvement. In the early 1990’s, Crinnion reported that the NCC had abandoned many of the traditional tools in favour of structures systems analysis techniques.

2.3.3 The Structured Systems Analysis Approach

Crinnion (1991) described the structured systems analysis approach as having five main aspects: a formalised set of tools and techniques, a standardised framework describing essential phases, a distinction between the physical and logical elements of a system, provision for a user role and finally, structure for the data to be used.

The framework of the structured system offered a series of stages with interlocking tools and models which provided cross-checking capabilities. In this early stage, the structured approach provided tight and consistent communications channels between elements and phases.

Crinnion asserted the structured approach removed a considerable degree of flexibility from the analyst and thus restricted the scope for creative solutions. This was seen, however, a worthwhile trade-off, sacrificing the “small possibility of a
Figure 2.1: An Hegelian History (from Boehm (2006))
quick, cheap, brilliantly intuitive solution” for the more probably stable and workable one. Crinnion compared this to a form of inbuilt risk-control.

Significant bureaucratic overheads however could impose burdensome time delays. Hanna (1995) quotes Cambridge Technology Partners (CTP)’s Vice President (Technology Services) Burt Rubenstein’s observation that traditional methods typically consumed six months in the requirements elicitations phase.

### 2.3.4 The Software Development Lifecycle (SDLC)

With the development of the traditional and then the structured approaches, a clear image emerged of a phased software (or system) development lifecycle. In its earliest descriptions, understandably, the notion was associated with the traditional or structured approaches.

The software development lifecycle (SDLC) became perhaps one of the most widely misunderstood notions in the field. Authors, lecturers and commentators frequently need to define and redefine the concept. Some adherents of radically different approaches have been known to simply equate the term SDLC with those earlier approaches which first gave it prominence.

The SDLC is neither a methodology, method nor a technique nor tool. It was simply a conceptual framework. It was no more, no less, than the notion that a software project (product), as for any project, has a lifespan, from inception to delivery, and possibly beyond, into maintenance, review and revision. It served, and will continue to serve, as a useful metric, a notion for the passage of ‘project-time’, and/or for the things that get done, irrespective of their relative weightings, sequencing or simultaneity. Writing about the much later Agile methods, Abrahamsson et al. (2003) made extensive use of the SDLC notion (as a metric for scope comparison) to provide a framework against which to examine and compare various Agile methods.

**Figure 2.2: The Waterfall Lifecycle (from Satzinger et al. (2009))**
2.3.4.1 The Waterfall

Figure 2.2 presents a typical conception of the waterfall SDLC. The phases are distinct and segregated.

There have been numerous variation of the classic waterfall presented over many years, but the basic principle remain one of distinct phases which are completed prior to commencement of the next.

Waterfall variants allowed for iteration between some stages or groups of stages. This allowed for some back checking and revision prior to further progress.

The ultimate iteration would be the ability to repeat the entire waterfall. And this notion is expressed in some significantly different lifecycle models.

2.3.5 Evolutionary Development

Evolutionary approaches build on the notion of exposing early versions of a project to user comment and refining the product in accordance with feedback (Sommerville 2001). Two modes of evolutionary development noted by Sommerville were; exploratory, which feels its way towards fruition with the cooperation of the customer, adding features as needs are identified, and throw-away-prototyping where lightweight experimental product is exposed to the user to gauge their reaction and refine an understanding of requirements.

Floyd (1984) distinguished between horizontal prototyping, in which a shallow level of development occurs across the entire scope of the project, and vertical, in which certain discrete functionalities or modules are developed in depth, prior to the decision to add to the side some further functionality or module. Nielsen (1994) neatly described evolutionary prototyping as horizontal prototyping and incremental prototyping as vertical prototyping.

Martin’s (1991) proposed Rapid Applications Development method was a distinctly Vertical approach in which the issue of project blow-out (due too often to volatile customer demands) was countered by defining each suite of deliverables as separate agreed projects. It could be argued that Martin presented analyst-designers with the tools to manage unruly stakeholders.

A large number of prototyping variants have been observed in the literature, but broadly, for the purposes of this review, their characteristics have been described.

2.3.6 Incremental Delivery

Though Avison and Fitzgerald (2006) report that Gilb was working in the UK on an evolutionary software design method which could be seen as a precursor to agility, Crinnion (1991) citing Gilb (1988), reports that Gilb’s call for ‘incremental deliveries’ was a response to a poor uptake of phased approaches to software engineering as observed in practice.

2.3.7 The Spiral

In 1988 Boehm proposed a particular iteration of the waterfall in which the entire process repeated, typically three or sometimes four times. This could be described in
a spiral, in which iterations move outward from the centre over time, typically clockwise. Figure 2.3 (above) presents the classic image of the Boehm spiral.

Boehm’s *bent*, iterated ‘waterfall’ contained four broad key steps which were repeated on each rotation. The first was planning which determined project objectives, alternatives and constraints.

Although the tight coupling of the standard structured approach had been described as a form of risk control, Boehm incorporated a distinct risk analysis phase which analysed alternatives with regard to their risk factors and established a measure of feasibility.

In the third, or engineering, phase a prototype of the product would be constructed and tested. The final stage of each spiral would be customer evaluation of the delivered product.

The Boehm Spiral has been a great influence, giving an iterative framework and a clear alternative to older more linear lifecycle conceptions. It has served as a watershed against which some other conceptions are measured.

![Figure 2.3: The Spiral Lifecycle (from Boehm (1988))](image-url)
2.3.7.1 WINWIN

In 1994 Boehm et al. (1994) explored a software negotiation tool which ran on unix workstations called WINWIN. The notion was that this groupware support tool and its accompanying approach could assist in reaching consensus on requirements by allowing developers and users access to a computer mediated common negotiation space. A decade after his iconic spiral, Boehm et al. reconciled WINWIN with the classic 1988 spiral (1998).

It was a testament to the longevity of the spiral in this volatile domain space that it could serve as the explanatory basis for a sophisticated group support system a decade after its initial publication.

2.3.8 Object-Orientation (O-O)

Avison and Fitzgerald (2006) described the complex origins of Object Oriented (O-O) as arising among numerous writers of the late 1980’s and 1990’s. These were traceable to ideas such as the capability based architecture (Browne and Smith 1984) which was later deployed in the agile community (Lahanas 2008). A more complete description might have mentioned concepts from a much earlier age in computing. Kay (1993) reported that, whilst struggling in the 1960’s to understand a modified form of Algol, known as Simula, so as to able to use it to procedurally control the sorts of objects found in Sutherland’s 1963 Sketchpad, he encountered (translated from Norwegian) strange usages of words such as activity and process. The prime notion was that “each Smalltalk object is a recursion on the entire possibilities of the computer. Thus its semantics are a bit like having thousands and thousands of computer [sic.] all hooked together by a very fast network.”

The breakthroughs towards object orientation inspired by this, which were first seen more widely in the Smalltalk-76 release, grew from a time before large scale debates of method and approach. Kay (1993) stated that, in creating the Smalltalk language at Zerox PARC, “we were actually trying for a qualitative shift in belief structures—a new Kuhnian paradigm in the same spirit as the invention of the printing press—and thus took highly extreme positions which almost forced these new styles to be invented”.

Sommerville (2001) lists the phases of the O-O approach as object-oriented analysis, object-oriented design and object oriented programming. The large phases were thus fewer and simpler in arrangement than in previous SDLCs, however each constituted a distinctly separable mode of practice (or profession) and contained within themselves numerous complex and interacting techniques and tools. It should be recognised that O-O methods postdated O-O programming, which in some early forms was implemented under a structured approach (Avison and Fitzgerald 2006).

Curiously, what Sketchpad had called masters and instances, Simula called activity and process, Kay went on to label his constructs objects and methods. It is curious to observe that Kay’s ‘new paradigm’, representing so radical a shift that later teaching texts (such as Satzinger et al. 2009) would simply divide the world into Structures and O-O (or as, process orientation and data orientation), had notions of structure and of process, of capability and even of activity. To deploy some of these terms (outside
the rigid strictures of UML) might to some modern O-O advocates, have smacked of near heresy.

Pierce (2002) described any attempt to describe Object Oriented programming in a list of features as futile. After a review of 239 academic papers, 88 of which listed O-O concepts, Deborah Armstrong (2006) identified 39 commonly mentioned O-O concepts. A literature analysis (including a count of occurrences, and incorporating taxonomic arguments) allowed Armstrong to isolate the top eight concepts which she labelled “quarks”. These, in descending order of occurrence in the literature reviewed by Armstrong, quarks were: inheritance, object, class, encapsulation (incorporating notions of information-hiding), method, message passing, Polymorphism and abstraction. Armstrong was keen to help identify a fundamental and consensually agreed set of O-O concepts to aid “developer learning by providing a common language and knowledge base.”

Perhaps unsurprisingly, as O-O grew from a coders’ perspective, it was not strongly linked to the broader user-perspective, such as adjusting a business process (as necessary) to take best effect of an O-O software system. Henderson-Sellers and Serour (2000) observed that whilst O-O methodologies offered good advice on software development, they had little to offer on deploying it into a business context. Echoing Jacobson, Henderson-Sellers and Serour emphasise that that ‘you need to know how to use it’, though while Jacobson was calling for a software development process to accompany the O-O toolset, Henderson-Sellers and Serour were referring to the end user’s business. To that end, they adopted the team building techniques of Baudoin and Hallowell (1996) towards proposing a framework called OPEN (Object-Oriented Process, Environment and Notation). The question remained, if Henderson-Sellers and Serour were offering techniques and frameworks to assist businesses reconstruct themselves to an O-O worldview, or, if they had even considered that such a system-centric view may result in changes to business processes, culture and strategic positioning, which were not initially requested, necessary nor desirable. If not applied with care and due attention to the clients’ initial motivation, there could perhaps have been some risk of putting the cart before the horse.

Whilst Henderson-Sellers (2000) declared that object technology had “succeeded rapidly” and “consequently succeed[ed] in building high quality software on time and within budget;” Potok et al. (1999) observed, across a survey of 19 commercial products, no productivity benefit in the O-O approach over that in the older procedural approaches.

It should be noted that Potok study was limited to the first two generations of O-O adoption, and the benefits of O-O’s promised reusability may not have been observable at that time. Potok et al. offered, in balance to this limitation, that productivity benefits under O-O’s promised by “simpler design” (unaffected by the same temporal issues as the reuse benefit) were also unobserved in their study. Likewise, they found no evidence of improvements as a result of O-O’s claimed “improvements” in design. They speculated that improvements in business process may be more likely to produce a productivity gain, and that the adoption of a new technology (such as O-O) would prompt such a process update. Undertaking a radical change in work practices and paradigm would necessitate a re-evaluation of
old processes which may have become stale. That a business process update of this kind can only be prompted the by adoption of O-O was not supported.

Further to criticisms of the heavy load imposed by the complex O-O environment, Joe Armstrong (2009) criticised the effect of the implicit baggage which accompanied the O-O programming approach saying, “You wanted a banana but what you got was a gorilla holding the banana and the entire jungle.”

Growing out of a perceived need to improve efficiency and the working experience of the software builder, O-O fundamentally shifted the paradigm to focus on self-contained code entities which could be built and tested separately and reused as required. To achieve this required a massive shift in analysis and design thinking which made O-O particularly challenging to learn. As Kuhn (1962) famously observed, revolutionary shifts of paradigm produce incommensurable points of view. Traditional practitioners found O-O challenging to accept, and new entrants to an O-O world could hardly be expected to perceive many benefits in what had come before.

If structured approaches were a reaction to chaos, and evolutionary approaches a reaction to phased bureaucracy, then O-O represented a fundamental shift in favour of the coders themselves, a technical approach on behalf of the technologists. The analytical overhead was, if refocussed, little reduced and there is scant evidence for any improvement in software productivity or reduction in failure rates. O-O continued to be difficult to grasp, difficult to apply and, perhaps most tellingly, difficult to teach. The need for simpler representations of the O-O weltgeist to use in both teaching and practice would produce suites of tools that would escape out from under the O-O umbrella.

2.3.8.1 Universal Modelling Language (UML)

To support the paradigm shift to O-O, and its extensive suite of new notions and modes of thinking, it proved useful to assemble, if not a universal expression of the approach, at least a common set of descriptive terms and diagramming techniques.

Rumbaugh had contributed a O-O analysis, Booch an O-O design and later Jacobson an O-O engineering. Rumbaugh and Booch formed the company Rational, which in 1995 acquired Jacobsen’s company Objectory. (Booch et al. 2000)

The Object Management Group (OMG) was established in 1989 (Satzinger et al. 2009, www.omg.org), as a consortium of (at that time) eleven significant corporations to establish standards for O-O systems and modelling. Vinci (2004) presented a road map history (Figure 2.4) of the Unified Modelling Language (UML) up to its 2003 form, which shows OMG adopting UML in 1997 from the UML Partners (another consortium lead by Booch, Rumbaugh and Jacobson, formed the previous year).

Under what Satzinger (2009) described as a divide and conquer approach, Object-Orientation requires that the analyst-designer master the use of UML’s multiple interrelated sets of diagramming techniques.
In 1998 Jacobson declared that with the rise of UML, being the ‘one modelling language’, the end of the ‘method wars’ was upon us (Jacobson et al. 1999). Avison and Fitzgerald (2006) were understandably hesitant to agree and pointed out that even given the significant advance made for O-O approaches; it might have been the result of a narrow perspective on Jacobson’s part, concentrating as it did on a suite of modelling tools and techniques rather than on full methods or even methodologies. To be faire, Jacobson added the caveat that an analyst-designer would, of course, need to know ‘how to use it’.

UML comprised a great many forms of diagram, but perhaps its most recognisable was the Use Case, if only because they eventually escaped from O-O and appeared back among the structured and ad hoc methods seen in practice (Satzinger et al. 2009).

Satzinger et al. (2009) reported that systems development starts with the identification of events which trigger use cases, elementary business processes. Ambler (2009) offered some advice for applying the use case diagram. He suggested asking “how the actors interact with the system to identify an initial set of use cases”, though it must be remembered that this ‘system’ has not yet been built or even described – curiously, this is an early stage (under O-O) of discerning quite what it should be. Each of these actors is joined, in the lightweight use case diagram, to the

Figure 2.4: A History of O-O (from Vinci (2004))

UML comprised a great many forms of diagram, but perhaps its most recognisable was the Use Case, if only because they eventually escaped from O-O and appeared back among the structured and ad hoc methods seen in practice (Satzinger et al. 2009).
use case, often to the specific actions that would occur within the system, as a result of the actors ‘request’ or trigger.

A generic user case diagram (Booch et al. 2000) is offered in Figure 2.5, showing a user triggering some system functions or processes so that it can act to fulfil some goal this primary actor needed to satisfy. The secondary figures on the right serve to “assist the system itself” (Cockburn 1998), by having input to some element of the use case, necessary to satisfy the primary user’s request.

The mechanism for describing discrete lumps of system functionality from the isolated user triggers remains mysterious. It seems to require considerable experience on the part of the analyst-designer. Ambler (2009) says of a slightly later stage that; “As I begin to notice similarities between use cases, or between actors, I start modelling the appropriate relationships between them” which left open the question of quite what similarities he noticed, or more importantly what relationships?

Krutchten (2004) advised the use of a scenario based analysis, where the analyst traced a ‘thread’ of sequential actions to defining use cases. It is not clear if there is any informing theory informing the identification of such threads, or simply common sense and the experience of the analyst. Krutchten went on to explicitly state that “the use-case model is a result of the requirements discipline.” Objects and classes were “most likely found by walking through the use cases.” (Krutchten 2004)

![Figure 2.5: A Generic Use Case (from Booch et al. (2000))](image)

Of course the system will comprise the functions of many use cases. Ambler (2009) describes three possible relationships between use cases, those of extends, includes, and inheritance. Krutchten (2004) taught that the use case model consisted of all use cases, which when somehow viewed together, described the “complete functionality” of the system. Although not any typical usage of the UML diagramming method, Figure 2.6 indicates this author’s conception of how multiple use cases, in some combination, should serve to inform the description of a system yet to be fully described or built. It should be noted that each actor is depicted in isolation from
their larger organisational context. Although the figure indicates numerous actors, there are no meaningful linkages between them.

**Figure 2.6:** Multiple Use Cases, overlayed to investigate the system

A more specialised UML diagram, the business (or essential) use case (Constantine and Lockwood 1999), is intended to capture intentions of user. Constantine and Lockwood describe this essential use case as a structured narrative of the stakeholders’ own terms. The function of this simple and technology-independent diagram was to describe one task. Whilst the notion of more carefully recording the users’ intentions (or motivations) in their own words is doubtless of great importance, the limitation to one task (however that might have been clearly identified) prevents a representation of the more complex interactions between tasks in the greater business process.

**Figure 2.7:** A Complex of Use Cases, inverted to investigate the network of users
Figure 2.7 (above) most certainly did not appear in the O-O literature. It was a speculative transformation of Figure 2.6, turned quite deliberately inside out to place the actors together. It illustrates that the fractured glimpses of the system given by the use cases are not easily seen in the context of the users network of interrelated tasks and actions. It must be noted that UML does contain notations for business processes (Booch et al., 2000); some primers however, such as Satzinger et al. (2009) and Ambler (2009) highlight the use case in particular as the first step. The deliberate perversion of the use case presented in Figure 2.7 is was intended to highlight the risks of putting ‘the cart before the horse’. To this author, it seemed perhaps some misreading had occurred, that the use case must surely be deployed after considerable primary analysis. Krutchten (2004) however, quite unambiguously called the use cases the ‘front-end activity’ (effectively equating it to the ‘requirements discipline’), the “common language for communication between the customer or user and the system developer,” and described use cases as “the bridge that unites requirements and design activities.”

In one of his famously tongue firmly in cheek critiques from the publication Queue, Bell (2004) proclaimed the risks of ‘Death by UML Fever’. A senior software developer with General Electric, Bell seems to have made it something of a mission to highlight hasty, inappropriate, and uninformed over-adoption of new trends in software development. His concerns in this respect were echoed in an entertaining keynote address by Gilb (2005) which predicted the continuation for 25 years of industrial addiction to buzzwords and flavour-of-the-month methods. Bell bemoaned the root cause of the hasty selection of techniques as being “a lack of practical experience” which he characterised as producing “both unrealistic expectation and misapplication of technology”.

Perhaps of themselves, the UML diagrams could not end the method wars, but when combined with an informing overall process (or methodology), it seemed possible.

2.3.8.2 Rational Unified Process (RUP)
A standard language for modelling of itself was insufficient for Jacobson (2000) and a process for its application was necessary. Vinci’s (2004) history map (Figure 2.4 above) for O-O showed how Jacobson’s work, combined with that of Rumbaugh and Booch, lead towards a product known as the Rational Unified Process (RUP) in 1998 later purchased by IBM.

It is interesting to note in RUP the use of concept of the worker (being a ‘work role’ or ‘the hat being worn’ (Krutchten 2004) associated with cohesive sets of activities most commonly performed by one person. As with the identification of tasks or of use cases, there did not seem to be an easily identifiable coherent theoretical basis informing the identification of workers or the cohesive sets of activities, suggesting yet another appeal to the ‘experience’ of the analyst-designer.

By the definition they developed and promoted, Avison and Fitzgerald (2006) would call RUP a methodology but reported that Jacobson would dislike the title. Jacobson et al. (1992) himself deemed methods in general to be little more than simple introductions to more mature (presumably bespoke and hand-crafted) work processes.
RUP is a significant approach, a highly developed application of the O-O concept, backed by a consortium of major industrial players, and it deserves considerably more extensive description that it has received here. The core notions for this review though, are that whilst RUP was a robust, flexible and somewhat prescriptive suite of iterative O-O flavoured methods, it retained a considerable need for the input of the analyst-designers’ personal experience. Further, it did not overcome some of the more fundamental issues of O-O, being a less than convincing treatment of the user’s connected network of business tasks, and the complexity and burdensome code-heaviness described for the O-O approach in general.

2.3.9 Agile (Lightweight) Methods

Crinnion (1991) details a case for incremental delivery of system components, developed in prototype form, to customers who may prefer a “‘flawed’ system [...] to no system at all”. Crinnion cites calls by Gilb (1988) and Rzevski (1984) for incremental delivery of prototype level system components to meet the demands of impatient customers. This early expression of an agile notion seems a compromise solution for a professional community incapable or unwilling to follow what methods and methodologies were available to them.

Abrahamsson et al. (2003) offered a somewhat Hegelian historical evolutionary map (that is, a deterministic plan of providence (Heidegger 1988)) for the nine agile methods they reviewed, here given as Figure 2.8.

In his broader roadmap, Boehm (2006) identifies issues of bureaucratic overhead under traditional structured approaches, human factors in object-orientation and rapid changes in risk-driven approaches as contributing to the rise of agile and hybridised methods. These are illustrated in the detail presented in Figure 2.9.

High system failure rates reported for 2009 by Crear (2009) as “the highest failure rate in over a decade” would not support any contention that the appearance of agile methods has made a significant impact on the overall success rate.

The so called Agile Manifestio, (Beck et al. 2001) declared four broad statements of relative valuation to which the seventeen ‘signatories’ adhered:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan.

The manifesto further set out twelve broad principles, including that “Working software is the primary measure of progress”. One might be tempted to voice the loosely inferable corollary, that: “comprehensive documentation impedes progress.”

Despite the publication of the manifesto in 2001, Abrahamsson et al. (2002) were able to state clearly in following year, that there was “no agreement” as to a definition of ‘agile’. In response to the perception that agile methods had largely evolved in practice, rather than in academia, and had thus not at that time, yet been extensively researched with sufficient academic rigour, Abrahamsson et al. (2003)
presented an overview of agile methods. Their intention was to rationalise and clarify a “dispersed field” which had “exploded during the last years and is not showing any signs of ceasing”. Nine different agile methods were investigated and compared, largely against the framework of the software development lifecycle (SDLC).

Abrahamsson et al. offered gentle criticisms that none of the nine agile methods they investigated were extensive across the SDLC nor were any of them precise. Those which covered more of the SDLC were too general or shallow (only three of the nine methods examined offered any concrete or prescriptive guidance). Those which concentrated on tighter phases of the SDLC tended to suffer from too narrow a focus and tended to lack any connection to other methods. Agile methods were therefore of severely limited applicability, either across the lifecycle or in combination with other methods or techniques used in other phases of the lifecycle. Only three of the methods which Abrahamsson identified as offering “concrete guidance” picked up at the requirements specification phase. In general, as with many other approaches, agile methods seemed to gloss over the early elicitation of requirements. In this, they offered little that was new, or would improve alignment of specifications to users needs. The agile philosophy rather, was to embrace the fluidity of user needs and deliver product component-wise. It remained unclear quite when (given the dynamic and fluid state of user requirements) a project ‘ends’; perhaps when some arbitrary completion date passes, some budgetary limit is reached or when customers and designer-coders alike collapse from mutual fatigue.

---

**Figure 2.8: Evolutionary Map of Agile Methods** (from Abrahamsson et al. (2003))
Figure 2.9: Detail from Boehm’s (2006) Hegelian History [emphasis added]

Delays in product delivery are anathema to the agile principle. Customers will be less than satisfied by lengthy delays (brought on by over commitment to extensive bureaucracy, documentation and formal phasing perhaps) and further, the longer a project runs, the greater the odds of a change in customer requirements – ensuring lower acceptance rates for the final product. The principles of agility might therefore, somewhat facetiously, be read to infer that if sufficient product can be delivered sufficiently quickly enough, then perhaps customers would be sufficiently satisfied before their minds change again, allowing the design team to ‘escape’.

Whilst, as with many works in the SE domain, agile techniques and principles are gathered under the title “methodology”, the lack of comprehensive scope, of prescription and, most importantly, of an identifiable ideological framework (aside from the somewhat mutable catechisms in the manifesto) deny them this title. At best the various agile “ways of doing” SE must be relegated to the status of methods.

Theunissen et al. (2003), themselves enthusiastic misusers of the term ‘methodology’, noted a vigorous debate between agile method proponents and supporters of what the agile community labels, big-design-upfront (BDUF) approaches. One crucial aspect of the debate was a perceived lack of documentation in agile methods, opening them up to accusations of non-compliance (or, at best, unverifiable compliance) with accepted software engineering standards (which may, for regulatory or organisational reasons, include documentation itself). Theunissen et al. (2003) cited the agile principle “working software over comprehensive documentation” as being apparently contradictory to the usual documentation
requirements for standards compliance. In an attempt to address this issue, Theunissen et al. proposed some guidelines by which various agile methods may incorporate processes to ensure compliance.

Agility may be said to have evolved from practice, in something of a methodological vacuum. It was not clearly identified, identifiable nor categorised in the academic taxonomy of software engineering approaches. Enthusiastic proponents, perhaps motivated by the promise of an escape from under the ‘burden’ of traditional “method talk” could be heard speaking of agility as an alternative to the SDLC, as though “SDLC” itself were the epitome of traditional structured approaches. Interestingly, acknowledged academic writers on agility, such as Theunissen et al. and Abrahamsson et al. (2003) required deployment of the notion of the lifecycle (in Theunissen’s case, under the title, “primary lifecycle processes”) to provide a comparative framework against which to describe their proposed compliance guidelines. Perhaps their writings served as a gentle rebuke to their more excitable agile colleagues.

It might be considered that agility, possessed as it is of many admirable qualities and apparently informed by numerous laudable (if flexible) principles, offers a way forward. Certainly it promises to deliver ‘something’ rather than ‘nothing’ to the stakeholder. Perhaps however, if agile developers were to consider themselves members of the software engineering community, and thus ‘engineers’ of some hue; they might wish to take a page from the writings of traditional civil engineers such as Beder (1998) who cautioned professional engineers against the dangers and risks of using the stakeholders’ world as the test-bench for prototyping and experimentation of novel designs. Using one's customers as beta-testers would not be received in the realm of 'real' engineers as a good engineering practice. It would likely have been unfair to have condemned agile practitioners as ‘beta-testing abusers’ of their customers, however much their sometimes vociferous rhetoric might have invited the notion.

Agility, as a concept may be seen as an expression of pragmatism, probably in response to frustrations with extant methods and their results (or lack thereof). It holds an important place in the pantheon of SE approaches and methods, but can hardly be labelled a systems analysis and design methodology of any kind. Its very existence serves this thesis’ argument as further evidence of the dysfunction of its antecedent methods and approaches. It yields, however, valuable lessons and concepts, such as the Extreme Programming (XP) practice of writing tests before writing code.

The agile approach however contained as much variety as such a loosely defined title might suggest it should. Despite the broad criticisms applied to them, some succeeding methods were described in terms which seemed to address the shortfalls of their predecessors. Cursory evidence suggested some moderating, if not ameliorating, influences overlaying the more radical agile rhetoric of the early 2000’s. Of particular interest to this current research was the most recent reported in Abrahamssons review, the Agile Modelling (AM) method (Ambler 2002).

AM, perhaps unusually for an agile method, supports the use of ‘sufficiently’ advanced models (sensitive to organisational culture) to support “acute design
needs”. The principles set out by Ambler (2009) include that models should include the later need to test (as is the case in XP). They further include that models facilitate communication with users (a critical element of agility) even if it requires the use of “pretty” diagrams Interestingly, Ambler offered a gently disparaging view of CASE tools in this regard). Crucially, Ambler views models as the way to view problem spaces and identify requirements. Consistent with the agile philosophy, Ambler supported quick and lightweight disposable diagrams which may be discarded once the particular aspect they explored had been satisfied.

Agility as an approach, particularly in its earlier forms, presented as the ‘angry man’, impatient to simply ‘get on with the job’. It was no methodology, nor any panacea to the woeful failure rates for computer systems. It did raise numerous valuable points particularly:

- The burden of bureaucratic formalism can add to customer dissatisfaction, especially with regard to slow delivery.
- Customers change their minds frequently.
- It is better to compose test criteria before cutting code.
- Models, particularly easily understandable lightweight ones, are crucial to identifying requirements and facilitating customer communications.

The more recent AM method in particular offered considerable promise as a launching point towards reunification and rationalisation.

The question remained: is agile rapidity the only response to customer fluidity and indecision? Perhaps a more sensitive, structured and theoretically informed method from the very earliest stages of requirement elicitation could assist customers to understand their own needs, and thus ease the burden on systems designers and builders whilst increasing customer involvement and acceptance rates.

2.4 Sidebands of the SDLC

A full notion of a system development lifecycle would include consideration of both the business ecosystem it would inhabit and of any changes to that ecosystem under design, and the user’s interface(s). Each of these can stand apart from SAD, each possessing something akin to analysis and design phases.

If these side-bands of the lifecycle are reflections or refractions of the ‘normal’ SDLC, then the literature may indicate reflections, or refractions of similar challenges (such as poor success rates, method breakup, poor requirement, poor communications and a lack of coherent theoretical base).

Perhaps the literature in these sidebands may have suggested possible vectors towards solution, not yet fully developed in the larger body of SAD literature. This review digressed a little, to consider some aspects of these sidebands.

2.4.1 Business Process Mapping/Modelling (BPM)

It has long been understood that control of a business organisation requires an understanding of its processes (Taylor 1911, Deming, 1982 and Juran 1988), which spawned a large variety of mapping concepts and techniques. If computerised
information systems were to be constructed (or perhaps evolved, depending on methodology) to thrive symbiotically within the ecosystem of a business process, then it was only ever logical to strive to understand that ecology.

Business Process Mapping (or Modelling) (BPM), though touched upon by SAD methods, evolved into a management-flavoured near stand-alone discipline. OMG’s (2010) UML modelling tools contained extensive techniques and diagrams for the modelling of business processes. There are grounds to consider that if an attempt at a new SAD method could at least generate useful BPMs, a worthwhile contribution would have been made.

If one of the notions highlighted in the literature as needing further attention was the user and his or her goals and intentions, there was scope to pause and consider a BPM-styled mark-up toolset which specifically aimed to capture details of that kind.

2.4.1.1 Distributed Intensionality (\(i^*\))

Yu’s model, \(i^*\) (‘eye-star’), meaning “distributed intensionality” [sic.], though initially conceived as a business-process engineering tool, frames processes as social activities between actors who depend upon each other for goals to be obtained and task performed (Yu and Mylopoulos 1994). It links organizational design decisions to strategic business reasoning, which approach can be applied to the formulation of requirements for a software project.

\(i^*\) incorporates two main diagrammatic tools, the Strategic Dependency model (SD) and the Strategic Rationale model (SR) (Mylopoulos et al. 1999).

In examining some activity stakeholders deem necessary, SD’s deal with five entity types: the actor entity represents a human, synthetic or organizational active agent capable of decision. ‘actor’ is loosely equivalent to the concept ‘role’, which invites confusion. The ‘task’ entity, which need not represent a complete system specification, is an artefact of a dependency relationship, definable between actors, ‘resources’ are entities without outstanding open issues, and may be called upon by actors. A goal is an assertional statement between actors in which the dependee is free to choose actions that will resolve the goal. The ‘softgoal’ entity whilst also an assertional statement, is qualitative, with no clear notion of satisfaction. Softgoals may, at best, be “sufficiently satisfied” (or ‘satisficed’). Softgoals behave like preference constraints to be optimised (Krishna et al. 2004). Softgoals represent typically qualitative non-functional requirement, such as ‘the software shall be learnable’. Figure 2.10 (below) is an example of an SD model, examining the activity, “provide training to volunteers”.

Whilst entities are described in dependency relations within the SD, actors’ internal intentions and rationale remain unexamined. For a more complete model of the internal workings of an actor, Yu proposed the SR model, drawing upon the earlier work of Chung in representing non-functional requirements (Chung 1993).

The SR model may be envisaged as the inside of an actor. It is drawn as a dashed ovoid, extending from the actor entity node. Figure 2.11 serves to illustrate the nature of an SR, representing the internal structure of some actors from the action “providing training to volunteers”.
Chapter 2 – Literature Review

Figure 2.10 An SD Model (from Yu (1999))

Figure 2.11: An SR Model (from Yu (1999))
Within this SR are defined links not unlike those in the SD, however these linkages are more specific. Tasks are decomposed via task-decomposition links indicating an ordinal ranking. Goals may be linked to other entities with Means-End links indicating tasks or resources required for satisfaction.

Softgoals have means-ends links that are qualified with an indicator of the degree to which the linked entity contributes towards satisficement. The contributions include make, break, help, hurt, positive, negative, and, or, unknown and equal (Mylopoulos et al. 1992). In Figure 2.11, a task is seen to provide a ‘somewhat positive’ contribution towards the softgoal “training content easy to use”, within the actor “Training Coordinator”.

SR diagrams may exist within SD diagrams for the same activity. When opened out, dependency linkages previously connected to the actor may then link to an element within that actor’s SR.

2.4.1.2 Business Process Modelling Notation (BPMN)

The Business Process Modelling Notation (BPMN) (White 2004) was produced to provide a common notation for BPM practice. The motivation of the BPMN Initiative (BPMNI) was to provide a ‘standardised bridge’ between business process design and implementation. It would seem that BPMN hoped to achieve a level of acceptance similar to that of UML.

It was clear that efforts to produce standardised BPM notations were driven by acceptance that communication between stakeholders and analyst-designers were crucial and that some common ground was necessary to even begin meaningful decision making [Chung et al. 2000].

2.4.1.3 Business Process Re-Engineering (BPR)

Business Process Re-Engineering (BPR) was a radical notion of information systems development which aimed to reconstruct entire business processes. It stemmed largely from the work of Davenport (1993) and Hammer and Champy (1993) who argued that “dramatic improvements” in key performance indicators such as speed, quality and cost could be achieved under radical rethinking and redesign of business processes. The perhaps ironically named Hammer (1990) famously advised businesses, “don’t automate, obliterate.” The notion gave techniques such as BPM and tools such as BPMN a purpose.

Grover (2001) observed that the tenets of BPR sometimes provided a convenient justifying pretext for some radical down-sizing practices popular in the 1990’s. It was noted that sometimes valuable skills and hard-to-replace skilled workers were lost in the radical push to cut costs. Without the skills to ensure production of salable quality goods, simple reduction of operational costs could ultimately damage profitability.

Unfortunately, although Avison and Fitzgerald (2006) report that BPR was adopted by senior managers enamoured of its claims (conceivably aware of the opportunities presented under radical change), successes were rare. Wysocki and DeMichielli (1997) reported failure rates for BPR initiatives as high as 70%, a figure curiously
reminiscent of the worst for software projects reported under the OASIG and Standish reports.

Avison and Fitzgerald (2006) presented the case that the very radical nature of pure BPR itself as described by Hammer et al. may have been a contributing factor in its poor success rate, but that it could be deployed in a more iterative and flexible manner. Grover (2001) associates a new more informed and careful approach to re-engineering with the principles of Total Quality Management (TQM).

### 2.4.1.4 ISO9001 Quality Management Systems

The notions of TQM appear to be reflected in the International Organisation for Standards (ISO) 9000 series of standards, covering Quality Management Systems for businesses and organisations ([www.iso.org](http://www.iso.org)). The ISO standard 9001(2000) *Quality Management Systems - Requirements* in particular specified that compliant companies will, amongst other things: Define their key business goals, Know their key business processes and have a documented systematic process-oriented description across their activities.

That such requirements have been adopted by the ISO group and incorporated into some regulatory frameworks suggested that any method capable of a BPM output could make a contribution if it proved effective and comparatively easy to use.

### 2.4.1.5 Enterprise Resource Planning (ERP)

An Enterprise Resource Planning (ERP) system is often a complete, greatly expensive and integrated replacement for a businesses entire ‘back end’ systems, replacing at a stroke legacy and manual processes (McNurlin et al. 2009). In some respects ERP might have been considered an ultimate form of Business Information System (BIS).

Though a topic of exhaustive research in its own right, ERP systems in this review served to highlight one of the extremes of information/computer systems designed to enhance business performance. Unfortunately, Buckhout *et al.* (1999) reported some dramatic non-successes in ERP implementation, with an average cost overrun of 179%, delays averaging 230%, a 35% project cancellation rate and only about 10% of ERP projects meeting time and budget targets. These figures compare unfavourably with even those cited at the outset of this literature review, and could have suggested the possibility of an inverse correlation between the size and complexity of a computer system and its chance of ‘success.’

### 2.4.1.6 Participatory Design

If radical re-engineering presented the risk of inappropriate or insufficiently informed application by senior executives and/or external consultants risks major costs to a business (Grover 2001), there might have been an argument for dispersing systems design further down into the corporate structure.

Belotti and Bly (1996) described the benefits of participatory design processes where the involvement of all stakeholders is hoped to enhance usability, suitability and ‘fit’.
Mumford and Weir (1979) had earlier written of the benefits of achieving a good ‘fit’ between employees’ work requirements (that which they must do) and their own expectations and needs within their work roles. In describing their ETHICS method, job satisfaction, it was argued, could itself yield measurable improvements to business outcomes.

Avgerou and Cornford (1993) presented five core arguments for the adoption of user participation in the analysis and design of computer systems; on moral & legal grounds, that participants are a valuable source of information, that participants will more readily learn the new system(s), that participants will acquire knowledge of the new system and that user participation will enhance the quality of analysis.

In balance, Avison and Fitzgerald (2006) observed some reactions to user participation and the Mumford’s ETHICS Method in particular. One reaction was to be uncertain that unskilled persons could do design properly. Another was that managers would not accept the loss of control concomitant with the empowering of mere end users in designing corporate infrastructural systems. Mumford countered these fears claiming the users skills in their specific job roles more than qualified them to describe the requirements of that job, and argued against micro-managerial practices to contest the second concern.

It would seem grounds existed to consider the users in the elicitation of system requirements; however a structured method could serve well to help justify their empowerment to senior management.

2.4.2 The Human Computer Interface (HCI)

Concentrating largely on human computer interactions and system interfaces, Platt (2007) made a deliberately irreverent case for ascribing widespread failure in software to ignorance and arrogance among designer-programmers, who presume to know better than their users what they would need to have built. Whilst Platt may be overstating the case, some corroboration may be seen in Linberg’s (1999) observation that system builders maintained high job satisfaction even during periods of appalling product failure.

Earlier literature suggests that consensus on the role and positioning of HCI within, outside, beside or even beyond the rest of systems development was not clear. More recent literature suggests that no consensus was ever achieved, at least not between the HCI and other computer science, programming and SAD activities.

Gruden (1989) argued convincingly against a uniformity of UIs, or ‘consistency’ as he called it, declaring the notion to be ‘unworkable’. Seemingly at odds with principles often held as in high regard in the HCI community, Gruden is actually calling for bespoke or contextual design (Beyer and Holtzblatt 1998). Gruden is quite comfortable with what he called ‘internal consistency’ but found no need to enforce any ‘external consistency’ between systems. The sense seemed to be that no universal answers were possible, so each UI should be ‘fitted’ to its system and its users’ work environment.

User centricity informed Nielsen (1994) when he codified ten heuristics (presented in Table 2.1 below) for usability, which have achieved near canonical status in the HCI
community. Interestingly though, he is at variance with Gruden and claims value in consistency, as it reduces the learning load on users, increases the likely accuracy of a users presumptions and reduces errors by allowing users to recognize a correct action from their prior experiences. Neilsen perceived that whilst, after Gruden, systems are not at all universal, the general cognitive behaviour of users however, may almost be.

<table>
<thead>
<tr>
<th>Visibility of system status</th>
<th>The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match between system and the real world</td>
<td>The system should speak the users’ language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.</td>
</tr>
<tr>
<td>User control and freedom</td>
<td>Users often choose system functions by mistake and will need a clearly marked &quot;emergency exit&quot; to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.</td>
</tr>
<tr>
<td>Consistency and standards</td>
<td>Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.</td>
</tr>
<tr>
<td>Error prevention</td>
<td>Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.</td>
</tr>
<tr>
<td>Recognition rather than recall</td>
<td>Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.</td>
</tr>
<tr>
<td>Flexibility and efficiency of use</td>
<td>Accelerators -- unseen by the novice user -- may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.</td>
</tr>
<tr>
<td>Aesthetic and minimalist design</td>
<td>Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.</td>
</tr>
<tr>
<td>Help users recognize, diagnose, and recover from errors</td>
<td>Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.</td>
</tr>
<tr>
<td>Help and documentation</td>
<td>Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.</td>
</tr>
</tbody>
</table>

**Table 2.1: Usability Heuristics** (from Nielsen (1994))

That heuristics were required had been recognised and Myers’ technical report (1993), sponsored by the Avionics lab of Wright Research and Development Center
[sic.], identified a number of specific reasons why human-computer interfaces were (and perhaps still are) so hard to design and to implement.

Crucial among the difficulties Myers identified in UI design are that designers cannot easily understand the tasks conducted by users, and that existing theories and guidelines (at least as they were in 1993) were not up to the task. Myers cites a 1986 report by Smith which identified and detailed as many as 944 different guidelines. The evidence was that the quality of interfaces was attributable more to the skill of the designer, than to any particular method or guideline they may have employed. Myers further declares that though all HCI guidelines (as at 1993) recommended iterative design, it is particularly difficult to conduct, because there were no clear metrics for assessing what needed to be fixed between iterations, nor if an applied change would address the issue(s).

The implementation (coding) of UIs is made difficult, according to Myers, because (among other factors), they must be programmed from the “inside-out”. By this, Myers refers to an architectural paradigm whereby sub-programs and sub-routines are called by the interface, rather than allowing the allowing the application be “in control”. The image this conjures would probably be instantly suggestive to an O-O designer of Use Cases and Object Programming.

Myers also found that the testing of interface software was difficult (due to the qualitative nature of success-failure measures), that the tools to assist in interface implementation were overly complex and that programmer’s reported difficulty with modular UI softwares.

Though a number of the technical issues identified by Myers may, arguably, have been addressed since 1993 (particularly by the use of O-O approaches perhaps), there is little evidence to suggest that the methodological issues have been resolved. There has been no evidence that his primary finding; designers’ poor the understanding of users’ tasks, has been satisfactorily resolved.

A growing perception that the ‘scientific’ machine side of system building was distinct from the ‘artistic’ human side was becoming apparent in the literature. Of the less than enthusiastic embrace given to formal methods by the HCI community at large, Wright et al. (1997) wrote that; “Within software engineering there is a predisposition to abstraction and top-down design. From the human sciences community there is a predisposition towards prototyping and iterative design.” The distinction illustrates something approaching a paradigmatic incompatibility.

In perhaps an atavistic expression of what for some may already have been out-dated ideas, Olsen (1998), a proponent of O-O & ‘functional’ design, aimed to address ‘perceived gaps’ in the education of computer graphics professionals.

Olsen saw the UI as, in effect, a specialised application of computer graphics and thus as a subset of the programming discipline. Indeed, Olsen referred to the ‘stepchild’ relationship between HCI and the more mainstream software discipline. Rather than recognising HCI and UI design as a stand-alone discipline per se, Olsen wrote about the “granularity of control” over projects and design between the fields of computer science & computer graphics. There is a sense of an almost political
manifesto in Olsen, a feeling that he wrote call to bring interface building to heel under its natural parent discipline – computer science.

Indicating perhaps a growing divide between those computer scientists who dabbled in interfaces (perhaps striving to retain control) and a growing sub-discipline of HCI, Thimbleby (1990) deemed UI to be a portal to a computer system’s functionality.

Thimbleby proposed a markedly more formal approach to the design of an interface but was careful to retain a statement of value for the creativity, thoughtfulness & effectiveness of a good UI.

Whilst striving to apply science in untangling the inherent complexity of HCI, he counselled the UI designer not to let formality obscure or reduce humanity or to reduce choices in design. In this Thimbleby is recognising, perhaps grudgingly, the inherent craft nature of something as non-deterministic as the interface between a machine and a human. Ultimately, for Thimbleby however, UI design could not succeed without the input, and perhaps the firm guiding hand, of computer science.

Sidestepping issues of control or independence, Thimbleby’s political position is quite clear. UI design should not merely play a supporting role to computer science; rather computer science should broaden its scope to encompass it.

Offering architectural abstraction as a starting point for UI design, Torres (2002) sets the design and development of UIs firmly within a larger development cycle (echoing the over-arching SDLC). In their workbook-styled teaching text, Shelly and Rosenblatt (2011) even go so far as to declare system output and UI design as the very first task in the design phase of the SDLC.

There are signs that as the HCI discipline grew in size and as a literary presence, it gained the confidence necessary to stand apart and speak for itself. The can be seen in Preece et al. (2002), where interaction design is portrayed with an architectural metaphor. The UI was no longer an oracular shrine where humans prostrated themselves before the computers controlling functions, but rather a place where human conduct their business and may access a near-invisible and servile computerised assistant. For Preece, it is a dialogue about spaces and human-to-human transactions.

This new rhetoric informed impassioned speeches against the encroachment of technical issues into the field which was vehemently intolerant of any attempt to proceduralise or prescribe the creation of UIs (personal conversations, Doctoral Consortium, OZCHI 2005, Canberra). Perhaps it had become unfashionable to suggest linkages between the creative craft of HCI and the formal procedural ‘computer sciences’ or to even speak of HCI and broader system design practices in the same venue.

Accepting the need to tailor a UI to the tasks of the user (client) group, Rosson and Carroll (2002) suggested the use of scenario based design, analysing the activities of the user by the use of tools such as the Hierarchic Task Analysis (HTA). Rosson and Carroll however make no mention of AT however, despite Bødker’s declarations of 1991; evidence supporting the notion that (as Myers (1993) had bemoaned nine years
earlier) existing theories and guidelines were simply not up to the task of designing UIs, and there was room for one to be developed or adopted.

Dix *et al.* (2004) produced an authoritative and encyclopaedic teaching text, widely used for teaching HCI and UI design (including by this author). The text champions the work of Neilsen (1994) as establishing the groundwork for a pure HCI discipline.

Whilst the text presents perhaps the most convenient and accessible ‘one-stop-shop’ for UI design tools, techniques and notions (largely of the type later labelled ‘classics’ by Lauesen (2005)) it should be noted that none of the tools they detail can operate from end-to-end of the HCI:UI lifecycle. Certainly none of them link or translate seamlessly (if, indeed, at all) into the broader system design lifecycle (SDLC) which (despite an increasing rhetoric of isolationism by HCI practitioners), still existed beside, if not around, the UI’s own lifecycle.

It was especially apparent when one tried to concatenate a subset of techniques and tools from Dix *et al.*’s collection into a linear sequence for practical application (such as for an undergraduate HCI design course), that there were considerable issues of scale, of language and of concept. There were multiple moments of transition and translation from one tool to the next where an analyst/designer would need to switch mental tracks and could be excused for missing some crucial element, or even simply giving up. In the extensive personal experience of this author in teaching HCI, neophyte UI designers were more puzzled and bewildered by the array of options than empowered and equipped.

The rhetoric of yet another encyclopaedic text by Lauesen (2005) suggests things may have come full circle. From the earlier notions that UI design should be an adjunct to the programmer’s practice, through the ‘stand-alone UI craft community’, Lauesen called for some kind of integrated expertise and a new mode of practice. Lauesen attempts to present a comprehensive review of HCI approaches but is presenting his own *virtual window method* throughout the book. To his credit, Lauesen was apparently aiming to bridge the “two worlds” of programming and HCI. He was all too aware that they each held themselves aloof as the premiere craft.

For Lauesen, the suite of tools (set out in collections such as that of Dix *et al.* (2004)) were *classic*, perhaps even *aged*, concepts. These include the themes of usability, heuristic evaluations, prototyping and iterative design.

Figure 2.12 (above) adapted from Lauesen, indicates how a traditional waterfall cannot apply to HCI. Lauesen goes on to echo Myers’ (1993) observation that, for UI especially, iterative design is difficult if not inappropriate.

That investigation of HCI required studies into the human and socio-economic condition were highlighted by Shneiderman and Plaisant (2005) who commented on the remarkable diversity of the HCI field. They listed such disciplines as sociology, anthropology, policy making and management.

Adopting a notion from telecommunications theory of Quality of Service (QoS) as a metaphor, Shneiderman and Plaisant sought to optimise an interface for speed of use, on the assumption that *time* was a user’s primary consumable and they
counselled designs which optimised the use of that finite resource. Their second priority was the avoidance of errors. As almost an afterthought, they endeavoured to reduce user frustration.

For Shneiderman and Plaisant designing a UI was about tuning it to meet ‘genuine human needs’ (if not actual satisfaction) thus, moving beyond what they called the ‘vague notion of user friendliness.’

The procedure set out by Shneiderman and Plaisant required an initial elicitation of user needs (broken into tasks and sub-tasks), though quite how this is achieved remained as indistinct as it had ever been. In the second step, the designer should establish technical reliability, thus winning user trust. The third step enforced standardisation, integration, consistency and portability and implementation must be conducted under the best practices of project management.

Benyon et al. (2005) echoed Preece in speaking of UIs in architectural metaphors and of the ‘information space.’ Benyon’s HCI is entirely human centred; accessible, usable and engaging. Above all it had to be enjoyable to use. This is quite opposite to the more mechanical design philosophy of Shneiderman and Plaisant (2005) in the same year.

Though this author initially pondered investigating the construction of new HCI and UI design methods and retains a profound interest in the field, the broader investigation of the literature indicated that a malaise lay deeper within approaches
Chapter 2 – Literature Review

to SAD, and was reflected in the reception and products of the methodologies and methods informed by those approaches. HCI and the construction of UIs remains a vitally important field of research and a crucial element in system usability, acceptance and success, but in the big picture is something of a subset of a broader, underperforming, body of practice. That HCI and the construction of UIs can exhibit some of the same symptoms is not surprising, therefore it is hoped that some rehabilitation or revitalisation of the broader issues may see some benefits to the more specific disciplines of this kind.

2.5 Reflections Thus Far

Thus far the literature had revealed a nebulous cloud of approaches, methods, techniques and tools. The historical trend had oscillated between rigid structure and a near absence of structure.

2.5.1 Of Methods and Methodologies

Arising from the literature reviewed there had been a noticeable fluidity of terminology. It proved convenient to briefly explore some definitions of the differences between a methodology and a method, especially as the terms seemed to be used with near equality.

Avison and Fitzgerald (2006) defined an information systems development methodology as a collection of procedures, techniques, tools and documentation tools; deployed in phases; under some philosophical view (the lack of which demotes the collected practices and tools to a ‘method’). Similarly, Gonzales-Perez and Henderson-Sellers (2005) defined a methodology as “the specification of the process to follow as well as the work products to be generated, plus consideration of the people and tools involved, during a software development effort.”

Crinnion (1991) two components: a set of tools techniques and model and some kind of overall framework indicating which tools are to be used at which stages of development. Maddison (1983) defined an information systems methodology as “a recommended set of philosophies, phases, procedures, rules, techniques, tools, documentation, management and training for developers of information systems.”

Whilst for Maddison (1983), a set of philosophies suffices, Avison and Fitzgerald (2006) require that the methodology has been developed under theories and assumptions consistent with a ‘philosophy’. They require that a methodology, to deserve the title as such, includes specific reference to its philosophy.

For Checkland (1981), a methodology was “a set of principles of method, which in any particular situation has to be reduced to a method uniquely suited to that particular situation.” Checkland later (1985) required that the development of information systems operate under an intellectual framework (philosophy), a methodology and be pertinent for a particular application. For Checkland, the intellectual framework (philosophy) guides and constrains analysis and inquiry by providing the structure to make sense of the world. Checkland’s methodology is the operationalisation of that intellectual framework.
Armed with these definitions the question could be posed any new structured and procedural ways of conducting SAD, as to its being a method, or a Methodology.

2.5.2 A Software Engineering Crisis? - Method Fatigue?

Avision and Fitzgerald (2006) characterised the 1990’s as the period of methodology reassessment. Citing issues including those of: no visible improvement in productivity, complexity, over-engineering (‘gilding-the-lily’ in their parlance), a high skills pre-requisite, inflexibility, goal displacement, insufficient focus of human issues, methodology adoption hurdles and an absence in systems improvements; they conclude that most of the promises of the great1980’s methodology boom have come to nothing.

In response to these dramatic disappointments, Avision and Fitzgerald (2006) observed the rise of ad hoc practices, whereby industry has returned to pre-methodology times and indulges in, at best hybridised, but more likely non-structured practices driven by a reliance on the skills and experiences of developers. Not surprisingly, this is a barrier to entry for new practitioners.

In a curiously counter-intuitive if not extreme reaction, characteristic perhaps of an anti-method backlash literature at its strongest, Truex et al. (2000) actually formalised an Amethodical Systems Development (ASD) approach. ASD rejected structure, sequence, control, rationality or any claim to universal applicability. Perhaps the software world’s equivalent to a ‘church of atheism’, ASD presents the theory of no theories.

2.5.2.1 Of Silver Bullets and Near Misses

In Fred Brooks’ widely cited landmark paper, "No Silver Bullet — Essence and Accident in Software Engineering" (Brooks 1986), he identified, against a history of rapid development in computer hardware, the need for at least one order-of-magnitude improvement in the productivity, reliability or simplicity of software. Brooks bemoaned that no single development in technology or its management showed likelihood to deliver such an improvement. He chose the deliberately mythical image of the silver-bullets, as popularly employed in fiction to kill werewolves, to make his point more poignant.

Brooks’ main argument was that a step towards radical improvement necessarily abandons any search for a ‘silver-bullet’ or cure-all solution, as belief in such things itself embeds a naive mindset. He called for incremental, systematic and stepwise improvement even though no single path forward would, or perhaps could, be identified.

After Aristotle, Brooks classifies impediments in technological advance as issues of essence (fundamental qualities of a technology itself) and of accident (current, but not inherent difficulties). For Brooks, the activity which results in the production of software is, most definitely, an art. Design of software is, as for art, a fundamentally creative expression of some otherwise hard to grasp message or state. By its very essence, no artistic endeavour, nor its tools and techniques, can be deterministic.
For Brooks, the primary issue in building software was, *essentially*, its specification, design and testing. He strongly argued that failure to grasp the concepts underlying what are essentially *conceptual* products, would invariably lead to poor results. For Brooks, issues of construction error paled in comparison to the *essential* truth that no silver-bullet could exist for software building, a complex task which would always be ‘hard’. Brooks argued that if no silver-bullets could exist for a discipline such as Physics, which (he claimed) could lean upon the surety that there was some natural ‘truth’ out there to be found; then it would be even less likely one could exist for a discipline such as Software Engineering which explores an entirely invisible and artificial problem space.

Crucially, Brooks observed that, as software exists to embody function, and as *required* function varies over time, building software would remain inherently hard, as targets move. The very intangibility of software could encourage almost casual requests for requirement change by clients that could not comprehend the costs of reconstruction (as they might for some tangible structure). It is this temporal pressure, to hit a movable target before it moves again, which seems (in the literature) to have been a driving factor behind several methodological innovations. Certainly, as discussed above, the agile approach and its precursors, aimed at speeding development of software.

Perhaps Brooks’ most powerful observation was the multi-faceted nature of software. Being insubstantial, software *defies* depiction in the manner of a physical three-dimensional artefact and all efforts to map it geometrically (through various abstraction and conceptual mapping techniques) simply obscure issues along other dimensions. Multiple, perhaps even incompatible graphs would be necessary, and Brooks observed that they’d often be non-planar and non-hierarchic.

Brooks went on to survey some innovations in software engineering as it stood in 1986 (including early moves towards O-O such as the classes in Simula-67). His chief mode of enquiry was to ask if ‘hopes for silver’ were revolutionary or incremental. Consistent with his argument, Brooks did not identify any ‘magical’ or ‘silver’ revolutionary solutions, but encouraged work along promising lines. The most promising line for Brooks was an early form of participatory or end-user design, whereby naïve users should be equipped with powerful workstations and software and ‘turned loose’.

The most important function of the software builder, Brooks concludes, is the “iterative extraction and refinement of the product requirements.” Since clients simply cannot specify their requirements completely and precisely, Brooks leans towards the notion of rapid prototyping, which was a prominent topic at the time he was writing.


Bell not only dismissed the notion that several techniques, technologies and approaches are silver, but was equally scathing of their rhetorical presentation *as*
silver. Bell identified several failed silver-bullets such as Object-Orientation, high-level software languages and Integrated Development Environments (IDEs) as ‘low-grade alloys’.

Of the silver candidate concepts current at the time of his writing, Bell drew attention to eXtensible Markup Language (XML), Web services and the Unified Modelling Language (UML). He sarcastically allowed them all the title ‘silver-bullet’ because of their “explicit value to both text and diagrams” their “power to shift the economics of software development” and their capacity to alter the focus of “long-established engineering disciplines.” Bell clearly equated the term ‘silver’ more with its sheen and appeal than its proven value.

Bell warns of the ‘reverence’ attached to silver-bullet trends; a ‘sanctity’ which precludes questioning and presumes licence to drop previous good practices. For Bell, there is no inherently magical value obtained simply by recording data in XML. It may be useful, but it does not ‘cure’ anything at all. Dangers lay not in the harm caused by silver-bullets, but in the hype and hysteria which accompanies them, leading to less effective adoption (at the cost of dropping the use of less appealing, but proven techniques). Web-services were criticised by Bell largely for widespread rhetorical adoption without any certainty of utility, extent or even workability.

Bell was particularly critical of UML, a topic he had previously addressed in a little more depth (Bell 2004). Not only does Bell warn that UML is no more of a panacea than any other silver-bullet and thus could divert attention and distract software engineers from good practice, but that it actually creates excessive and pointless extra work. Bell dismisses UML Use-Case diagrams for their lack of contextual detail. He saw them as increasing the need for discussion and consensus forming to be of any value, whilst paradoxically reducing analysts’ perception of the need for any such “mind-melds” largely because of their silver-bullet status. Bell was scathing of the ‘primitive’ UML Use-Case diagram, claiming it was disconnected from reality and could “rob otherwise intelligent people of their common sense.”

Bell’s chief point was not to dismiss the particular trends of the time when we wrote, but to warn that applying any form of reverence to supposed silver-bullet solutions was actually harmful, bringing more pain than ‘simply’ using tried and true engineering practices. He proposes a return to Transylvania (legendary home of the fictional silver-phobic monsters Brooks (1986) had referred to) where one could hire werewolves to work on software projects, secure in the knowledge that they’d be wary of flying silver-bullets, unlike experts elsewhere who seemed unhealthily addicted to them.

The next year, to celebrate 20 years after Brooks’ landmark article, Mancl, Fraser and Opdyke (2007), (each from well known industrial organisations) offered a brief retrospective of Brooks’ 1986 paper to introduce a workshop on the theme. Little concrete evidence arises in this brief call, but Mancl et al. do maintain that the general malaise identified by Brooks 20 years earlier was still in force. The industry was still obsessed with the search for a ‘magic technology’ to provide an order-of-magnitude improvement. They called for a review to test if any of Brooks’ ‘promising roads’ had delivered since 1986 and to identify new paths to pursue.
In an irreverent moment during the OOPLSA’07 ‘Silver Bullet’ panel report (Brooks et al. 2007), Fraser stated that “OO is a dangerous and evil idea. But I have overcome it without much difficulty. It has many good ideas, but no one does it! Most of the world has no idea.” and further, of ‘Silver Bullet tools’ in general, that they are “tools that actually increase complexity”.

As raised previously in this review, Gilb (2005) foreshadowed for the coming 25 years no positive change away from a culture of chasing buzzword flavour-of-the-month methods. Perhaps the lessons of Brooks’ careful analysis, as feared by Bell, would indeed be ignored. If it had become a general perception of software development tools, techniques and methods that they were rife with over-promise and often constituted little more than trends and fads, how would industrial practice respond?

### 2.5.2.2 Poor Industrial Uptake

In 1988 Gilb had reported a poor uptake of phased software development tools and method in practice. In 2005 Kerkow et al. conducted a review of software development processes by small to medium enterprises in Germany and reported that “The current software process does not follow a procedure found in the literature, nor is there an underlying process model. It has grown through the course of the years from own experience, ideas, and solutions. Normally the process does not employ SE terminology.”

Taken together with Avison and Fitzgerald’s (2006) commentaries (cited above), these reinforced an undercurrent in the literature which gave voice to a reality rarely spoken aloud; that methods tend not to survive contact with industrial practice intact.

### 2.5.2.3 Method Engineering – The End of Method?

Perhaps the most structured response to a crisis of faith in software engineering methods has been the rise of various flavours of method engineering. Under the suggested *sub rosa* of a keynote address to a deliberately provocative workshop entitled *Advocatus Diaboli* (ENASE 2006, Erfurt, Germany, the 2007 follow up of which was co-chaired by this author in Barcelona, Spain), Brian Henderson-Sellers (2006) declared the ‘end of method’ (*sotto voce*).

Though reticent to commit such deliberately over-provocative statements to print, Henderson-Sellers was prepared to write that, “method engineering promotes the construction of methodologies for information systems development by selecting and assembling method fragments from a pre-existing repository.”

### 2.5.3 Is It All About Requirements?

As seen above, Brooks (1986) argued that the most important function of a software builder is the extraction and refinement of product requirements. If poor requirements are a crucial element in low software project success rates, and these are at least partially attributable to method failures and poor communications between phases, analysts, designers and users (itself another commonly cited cause of poor success) then there may have been a case for concentrating on a lightweight, readily learnable, methodologically flexible approach, based on a single theory.
(which addresses user activity) and which concentrates on the identification of requirements.

Kotonya and Sommerville (1998) identified four crucial aspects to the elicitation of requirements; an understanding of the application domain, an understanding of the problem at hand, an understanding of the organisation or business which is facing the problem, and an understanding of the needs of the stakeholders and applicable constraints. A thorough embedding in the context and situation, involving all players and their interactions was indicated.

Certainly a concentration upon requirements survived the method backlash. The work of Lamsweerde (2009) in linking goal based requirements to specifications (via the near-ubiquitous UML), the continuing work of the Mylopoulos group and other more formal approaches, suggested the requirement was far from dead.

2.5.3.1 Functional and Non-Functional Requirements

Whilst opinion, theory, technique and method of capturing requirements varied considerably, some traction was obtained in classifying them. The Atlantic Systems Guild’s Volere Requirements Template (www.volere.co.uk), championed by Robertson and Robertson (1999), provided a taxonomy for requirements covering the entire SDLC, inclusive of the earliest feasibility plans and management of the software project.

An important divide in the taxonomy of requirements, which pre-dated Volere, was that of the Functional and so-called, Non-Functional Requirements (NFRs). For Robertson and Robertson, a functional requirement was something a system (or product) must do, whilst an NFR set out a more qualitative property the system (product) must have. In the Volere template, the behaviour of the UI (particularly its classic qualities of usability and learnability etc) were quite firmly ‘non-functional’.

Chung et al. (2000) produced a series of papers and a text detailing a largely business-process-modelling inspired suite of models and analytical tools to specifically address the issue of non-functional systems requirements.

2.5.4 Turning Requirements into Specifications

Buchi and Weck (1997) argued the need for grey-box specifications. They argued that the highly abstract black-box specification (inputs and outputs only) were not sufficient for interactive components such as those in a UI.

The other extreme, white-box specification, would be verbose in the extreme and would hinder re-engineering and substitution because of the very prescriptive detail it provides.

Grey-box specifications therefore, were identified as a possible complimentary solution, able to provide sufficient detail for “those parts of the implementation that are relevant to the client programmers in a way that does not fix any unnecessary details.”
Chapter 2 – Literature Review

2.6 The Need for a Theory
In the ‘sideband’ of HCI and user interfaces UI, where fractured processes, mismatching techniques and communications breakdowns were as rife as in the wider domain of systems analysis and design; there had been numerous calls for the identification and deployment of a single coherent informing theory.

Kaptelinin (1996) declared that a lack of an adequate theory of human-computer interaction had hindered development in the HCI field, relative to that in the technical field and earlier, Myers (1993) had observed that theories and guidelines in HCI were insufficient.

Hasan (1998) had observed that IS was a discipline in search of an identity and a theoretical basis. Her suggestion was that perhaps AT could serve in that capacity.

2.6.1 Towards Activity Theory
Maciaszek drew the production-centric distinction between ‘business rules’ – a functional requirement describing an ‘always on’ (invariant) aspect of the system, and ‘constraint statements’ which defined restrictions on system behaviour or the production process (Maciaszek and Liong 2005). Whilst this distinction may not have strictly addressed stakeholder preference, it allowed acceptance or rejection on those grounds.

Stakeholders generated their own notations and terminologies, complicating the business of capturing such details (Sommerville 1998). This difficulty informed approaches to requirement elicitation that were sensitive to consistency and viewpoint. It was necessary therefore to use a systematic approach when capturing requirements. A significant risk of failure existed in marginalizing the stakeholder’s softer objectives, despite their inherent messiness.

One approach to identifying stakeholder preferences adopts a goal-orientation and asked ‘what does the stakeholder want to achieve’. Goal formulations expressed intended system properties (Lamsweerde 2001). Goal-centrism offered stability as top-level goals are often invariant under decomposition, and facilitate back-tracking when re-design issues arise.

Shifting to process orientation necessitated consideration of the software in its environment. It became necessary to identify those active elements that had choice (Lamsweerde 2001). Such active elements (actors) were the loci for the formulation of goals and preferences. Process-centrism thus inferred examination of actor goals, motivations (intentions), dependencies and cultural-historical activities.

Constantine and Lockwood (1999) observed the shift of focus from technology to people under user-centric design. To achieve a marked improvement in tool usability he said, “it is not users who must be understood, but usage.” Taking the notion further, Räsänen and Nyce (2006) argued strongly for a re-deployment of anthropology in understanding HCI partially because equating interaction for context can skew the focus to individual actors (users) when we should have been studying “larger” socio-structural processes involving both individuals and their technologies.
A rough taxonomy of these changing centrisms and foci might be given as:

- Product centric
- Process centric
- Goal centric
- User centric
  - Intention-Dependency centric
  - Motive-Activity centric

Perhaps the most widely studied and respected motive-activity centric approaches to the issue of requirement elicitation were based upon the psycho-social principles of AT. The history, development, principles and application of this theory are discussed in considerable detail later in the following sections.

Kuutti (1995) observed that the first interest in AT seen among the computer sciences was seen in the HCI community, where it had been viewed as a useful framework.

Writing in the domain of the requirements engineer (RE), Goguen (1993) argued strongly that social issues informed the need for systems, and thus their requirements. Further, Goguen observed that the very elicitation of requirements by engineers of their customers was inherently social. Whilst most traditional RE techniques had actively ignores social issues, Goguen was unhappy with the early psycho-social and normative techniques. He issued a call for the adoption of ethnomethodological techniques, which could get behind users’ own ‘myths’ (by which they misunderstood their own needs and goals). Whilst AT might have been classifiably ‘psycho-social’, from its inception it sought to disentangle the outcomes of human activity from the ‘objectives’ of those engaged in those activities.

Ditsa (2003) argued the need to move away from theoretical bases which were limited to the individual and their technology, and towards an awareness of the social context in which the technology was deployed. Ditsa advocated the use of the AT as its concepts would permit an holistic approach into information systems research. Ditsa did not specifically call for AT to inform the design of systems, but certainly argued strongly for its use in researching systems themselves. Martins (2007) however did declare that AT could offer a basis for software engineers which would address the ‘gap’ between technical and social issues.

It had been commented that AT held considerable promise for the elicitation of requirements for use in system design (Martins and Daltrini 1999), especially for interface design, however most instantiations to date had not delivered clearly systematic nor prescriptive methods for widespread adoption. In a blog entry titled “Activity theory and User-Centred Design”, Otwell (2005) bemoans the lack of prescriptive method for applying AT; saying “Activity Theory seems to almost defy practical application”. Otwell specifically cited Collins’ Activity Centered Design (ACD) work of 2002 as offering no concrete example of AT in use for design.

One notable exception to these criticisms was the Usability Evaluation Method proposed by Vrazalic (2004), which sought to improve the quality of systems (at
least in terms of their usability) by deploying AT to structure well-founded and systematic test criteria.

### 2.6.2 First Generation AT

Exploring the construction of human consciousness, Vygotsky (1978) proposed that humans conceive actions upon an internalised plane of reality. The actions of the human actor (subject) upon objective reality occur via various mediating tools (both physical and psychological). The current conception of this is shown in Figure 2.13.

The object is a focus for the Activity and directs the things that are done (Uden et al. 2008), irrespective of is resemblance to the final outcome. The object is informed by culture and history (being thus, subjective). In an organisational business setting, the sanctioned and even prescribed activities of subjects within that organisation would be primarily informed by the motivations and intents of the business itself.

![Figure 2.13: AT conception (from Vygotsky (1978))](image)

### 2.6.3 Second Generation AT

Leont’ev (1978) focussed upon specific activities and proposed a hierarchic model of components as shown in Figure 2.14. He proposed that all collective activities are directed to a single object. Any given activity has a motive. Within that motive are goals oriented Actions. At the base level, atomic operations are taken depending upon prevailing conditions. Leont’ev’s conception was a powerful and dynamic vision which encompassed the notion of components ‘sliding’ to another level (typically upwards) as the Subject devotes more cognitive attention upon them in the face of some unforeseen complication. Downwards ‘slides’ may arise as subjects become more familiar with actions and they devolve into near autonomic operations.

![Figure 2.14: AT hierarchy (from Leont’ev (1978))](image)

Martins (2007) drew out a crucial characteristic of this cognitive-involvement notion; that an Action is always consciously planned (oriented towards fulfilment of its goal), whilst an operation is an atomic stimulus-response to some environmental condition(s). The infamously difficulty slipperiness of AT may arise when, due to some unforeseen factors, combinations of conditions may require a more planned response, elevating the doing to the status of Action as a coping goal is formulated.
Also, an action may fall to the status of operation if it is sufficiently practiced and the trigger conditions become sufficiently regular and predictable.

2.6.4 Third Generation AT

Interest turned more to the role of the people engaged in the Activity, and Yrjo Engström developed a conceptual matrix, Figure 2.15, which expands on Vygotsky’s earlier work. The Subject: is the person or sub-group whose point of view is analysed. The relationships between these socio-cultural nodes are defined in the Division of Labour node and also in the rules node which contains social norms, regulations, paradigmatic procedures and other constraints.

The socio-contextual notion extends to encompass the relationships and frictions between neighbouring and inter-related Activities such as those described by Kuutti (Kuutti 1991). Consider, that the outcome of one Activity may become a tool or rule of another. One activity may determine the subject of another and so on. (Vrazalic 2004)

![Figure 2.15: AT matrix (Engström (1987))](image)

The reader should not feel satisfied that this brief sketch has done justice to the AT domain. Engström’s matrix is a powerful conceptual system whose applications are widespread and much debated in a variety of fields.

According to AT precepts, an activity is the unit of analysis, being motive driven, conducted by a Subject with the assistance of some tools and producing some outcome which might resemble the originally intended object. Beneath that were goal driven actions and further still, atomic level conditionally driven operations.

2.6.5 AT’s Historical Dimension

As described by Kaptelinin (1996), activities evolve constantly across time and any shifts in culture that might be described by that history. Kaptelinin understood activities to be temporally dynamic and history is offered as a basis for classification. Where Kaptelinin saw richness and complexity developing across an activity by the shifting cultural-historical trends around it, Engström et al. (1999) emphasised a social dimension. It might be argued that Engström’s many viewpoints and voices were the medium through which Kaptelinin’s cultural-historical developments were expressed and given form.

In an organisational business context, where subjects are engaged to conduct largely prescribed tasks towards a declared business goal, one would no less expect to find rich complexities arising from the variety of individuals, acting as subjects across the life of the activity.
To consider what form might best be described for a tool or toolset, to facilitate an organisational activity though, it may be best to operate within a single historical context, informed by the consensual culture of the workplace. In this regard at least, some of the deeper contributions of AT might be set aside for a time. Specific tools, designed to facilitate specific purposes, might best be described against as stable a context as is reasonable.

It is observed in the literature of the Social Construction of Technology (SCOT) (Pinch 1986) that a tool or technology comprises both the artefact and the cultural-theoretical context that informs it. Whilst it might be of considerable academic interest to explore the changes in the meaning and identity of a tool as theory culture and history sweep over it through the eyes of both SCOT and AT, it is beyond the scope of this current research. This study seeks to investigate possible means by which the analysis and design of tools for use in the here-and-now might be informed under a consistent theory. Though AT offers dimensions beyond considerations purely of the here-and-now, it might not be unreasonable in this study, to consider setting aside some of its historical flexibility and explanatory potential for the time being.

### 2.6.6 Engström’s Four AT Contradictions

Engström (1987) describes four levels of contradictions that arise within systems of Activities. Consistent with the notions of the first generation of AT, which observes tensions between object and outcome, a first level contradiction arises between the use and exchange values within each node of the Engström triangular matrix. These arise from contradictions in how the activity’s component actions interract. Different Subjects engaged (presumably at different times) in an activity might be motivated by differing sets of action goals. Engström observes how these primary conflicts give rise to instabilities and thus to changes and developments which occur over time.

The second layer of contradiction may arise between the nodes within an activity. For example, some tool might not entirely conform to a rule and/or some other similar internodal inconsistency.

The historical dimension of AT can give rise to a third layer of contradiction when there is a tension between an activity and some later reconfigured variant of it. Such variations may arise to accommodate temporal shifts in motivation, new tools, new communities and the like.

The fourth level of contradiction is of profound interest. It arises between an activity and other neighbouring Activities between which outcomes may be exchanged. The reader is invited to recall that the outcome of one activity might inform, influence or provide an element of another, such as a subject, tool, rule etc.

Although by way of example or case study, AT literature often addresses a single Activity and its components and tensions (remaining cogniscent of cultural-historical context and temporal variations), the recognition by Engström of fourth level, inter-activity, contradictions allows for consideration of networked complexes of activities, linked by their transacted outcomes.
2.6.7 More Recently...

Leon’tev was the student of Vygotsky (who died at a relatively young age with his work unfinished (Vrazalic 2004)), and as a result the two bodies of work are typically presented as generationally related (Engström 2001). However, they have been described as constituting two somewhat different frameworks altogether (Kaptelinin and Nardi 2006), in which the Vygotskyan tradition pursues individual development, and the ‘Leon’tevian’ with communal work. (Wilson 2006) However, both ‘schools’ (if one accepts they exist as separable traditions) are deeply wedded to investigation of the relation between intent-driven actors and the things they do (or at least, attempt to do).

Leon’tev (1978) proposed that all collective activities are directed to a single object. He specified however that ‘object’ drives activities, whilst the hierarchically lower actions and operations have less deeply involving drivers. Twenty years later, Bardram (1998) emphatically declared that it is the breakdowns, failures and problems which ‘invariably’ occur in co-operative work situations (or, he almost reluctantly allows, deliberate ‘shifts of focus’), which give rise to activity-driving objects by creating the need for mutually constructed solutions.

Uden et al. (2008) clarified that an activity may lose its ‘motive’ and thus be demoted (down through Leon’tev’s hierarchy) to an action, once the problem or blockage is resolved. The actors’ doings would become automated or ‘frozen’ into some form of regularity, which could be driven by a mutually understood goal. This was a particularly important observation, as it carried the unspoken inference that the ‘demotion’ of activities to actions may accompany, indicate or even bring about the resolution of problems, failures and blockages. For this study, it suggested a vector of inquiry, to try and solve problems thereby relieving groups of the need for effort (perhaps reducing what might be deemed some kind of involvement or cognitive load) in the conduct of their normal tasks, thus freeing their mental and emotional resources for more creative and innovative ends.

For Uden et al. (2008) however, it seems to have been the suggestion by Baerentsen and Trettvik (2002) that the reverse may be true that was exciting. Rather than relieving problem states down into mundane actions through resolution of blockages, Uden seemed particularly interested by the notion of elevating mundane workaday systems into ‘living’ and ‘dynamic’ structures able to accommodate organically flexible and dynamic tasks. Her work involved informing design notions for flexible web application interfaces to permit richly dynamic navigational activities. The inference is hard to avoid, that Uden was not wedded to the Bardramian notion that activities must accompany problems, but rather saw an opportunity for AT to inform designs which would facilitate complex cooperative work that might otherwise have given rise to blockages. Rather than freeing mental and emotional resources, Uden seemed to want to skip ahead to facilitating creative and innovative output, perhaps without directly addressing the mundane issues that were constraining.

Despite having forged a strong linkage between activity and cooperative work blockages (placing him squarely in the ‘Leon’tevian cooperative camp’, if indeed it exists), Bardram (1998) carefully describes three forms of collaborative activity: Coordinated work which occurs during routine activity; co-operative work which arises
when there is a blockage that gives rise to a common object; and co-constructive work which involves somewhat introspective and reflexive self examination by the group and may yield altered conceptions of object.

Bardram (1998) offered as a classifying heuristic, that a condition (the driver of an operation) is is formed somewhat ‘unconsciously’ outside of any awareness of a common objective; that a goal (the driver of an action) arises when a common object has become stabilised but, that an object proper (driver of an activity) forms only when there some blockage requires cooperative work towards resolution (or, just possibly, when there has been a deliberate ‘shift of focus). Whilst Bardram’s stated 1989 agenda was to highlight the importance of breakdowns, we must not forget that they are the sole cause of activities. Bardram clarified a useful set of distinctions which may assist an analyst in distinguishing operations from actions from activities. Whilst, as discussed elsewhere in this study (notably section 2.8) there are precedents for considering entire sets of activities (or ‘systems of activities’ as writers such as Uden et al. (2008) term them), Bardram’s clarifications unfortunately do not accommodate or describe that higher abstraction.

Bannon and Bødker (1991) set out a critique of what they deemed ‘limitations’ in the cognitive science tradition underlying much of the HCI literature at that time. They are somewhat disparaging of ‘traditional’ task analyses and approaches. Importantly, Bannon and Bødker (1991) question a call forcialization within the HCI field and between HCI and other aspects of system analysis and design (Draper and Norman 1984, cited in Bannon and Bødker (1991)). Bannon and Bødker caution against such separation because that which is being designed is far greater and more deeply integrated in the collective work environment that such approaches may be equipped to comprehend and accommodate.

Nardi (2005) and Kaptelinin (2005), at first individually, then collectively in a text constructed largely from their 2005 articles (Kaptelinin and Nardi 2006) worked hard to differentiate object from motive. Using the Russian terms objekt and predmet (although they have only very subtly different meanings in Russian according to Kaptelinin) they deployed objekt to be (mostly) material things existing outside the mind and predmet to be the “target or content of a thought or an action” (Ozhegov 1982, cited in Kaptelinin 2005). The distinction is perhaps rather subtle for an English speaker, not unlike that between intention and intension which was the topic of an interesting exchange between this author and Eric Yu, creator of the *i* notation (pers. conv at the University of Wollongong 2010). Curiously echoing that similarity, the differences between objekt and predmet serve for Nardi and Kaptelinin to highlight the role of an actors’ intention within an activity system. When they critiqued other approaches literature in their 2006 text, Kaptelinin and Nardi choose to (perhaps somewhat unkindly) point out that actor-network theory cannot avoid consideration of actors with intentions. (Walls 2009)

Kaptelinin (2005) carefully differentiated Leon’tev from Engström according to their perspectives on the ‘object of activity’. In this way, Kaptelinin declared Leon’tev’s work to reside more in the realm of the individual, of psychology, and of the desired outcome (predmet); whilst Engström held activities to be communal, to be concerned with produced outcomes (objekt), and to be more widely deployed in the field of
organisational change. If Kaptelinin’s distinction holds true, then perhaps this current study is more closely aligned with the work of Engström than of Leon’tev. It is not clear if Wilson (2006) would have considered this to be yet a further ‘split’ in the AT community or somehow symptomatic of the Vygotsky-Leon’tev divide he described.

Serving as a compliment to Kaptelinin’s 2005 paper, Nardi’s paper “Objects of Desire: Power and Passion in Collaborative Activity” (2005) highlighted the more emotive aspects of group activity. If an activity is driven by a communal object, created through some problem or ‘shift of focus’ (Bardram 1998), then Nardi is concerned with how that community will reach a concensual agreement on what that object is. It is interesting to observe that Nardi describes Leon’tev’s concession that a ‘need or desire’ standing behind every activity as ‘appearing’ to have been inspired by Vygotsky, and it is not inconsistent with Wilsons (2006) description of ‘two schools’.

Nardi (2005) carefully makes the case for objects, which by definition have a singular relation with an activity, to themselves be informed by multiple motives. Nardi stops short of declaratively associating motive with predmet, but does clearly split object from motive; a distinction less clear in some other works (such as that of Uden et al. 2006).

Nardi (2005) deployed an extensive case study to demonstrate how AT assisted in understanding the creation of object from multiple motives. She carefully distinguishes her use of the terms such as ‘desire’ and ‘emotion’ to avoid ‘reverting’ to discussion of ‘inner states’ (another ‘political’ positioning in Wilson’s (2009) ‘two schools’ dynamic?) but rather, is concerned with “the social nature of needs and desires as they are expressed in an activity system”.

Though their text has been critiqued by Walls (2009) as being somewhat overly confrontational, framed as it often is to ‘strike a blow’ against some well established Cognitive Science approaches to HCI, it did serve to disambiguate these important notions of ‘object’ and probably managed to put some non-Russian-speaking AT thinkers ‘on track’.

Kaptelinin and Nardi (2006), in exploring the social construction of objects under (possibly) multiple motives, list four possible criteria for “successful” objects: balance, which indicates that the various motives have been represented; inspiration, which indicates that the object is not just feasible but provides group motivation; stability, which indicates a temporal invariance of object; and finally flexibility, which permits change in an object to avoid obsolescence.

The observations, clarifications and heuristics offered by Nardi and Kaptelinnin (2005, 2006) certainly offered some guidance for observers and analysts of group activity. It might be argued that they set forth powerful heuristics to assist groups to resolve objects. In combination with Bardram’s earlier work (1989) the beginnings of a framework could be glimpsed which might assist in clearer understanding of objects and the activities they drive. It might be arguable that some navigational guidance through Leon’tev’s layers arises when Bardram’s definitions (1989) are linked with Nardi and Kaptelinnin’s clarifications of ‘object’ (2006). Certainly however, no distinct, declarative or prescriptive technique for had been declared.

53
Chapter 2 – Literature Review

Acknowledging that the idea was not new (though seemingly unaware of the relevant literature in the fields of history, philosophy and sociology of science and technology (Merton 1938, Kuhn 1962, Schuster & Yeo 1986, Pinch 1986)), Bannon and Bødker (1991) remind us that an artefact is an embodiment of theory. Any suggestion that a tool (as a Vygotskyan mediator) may be deliberately assembled in accordance with a motive (Brown, 2004, 2005, 2006a,b), much as the object itself is ‘constructed’ (Kaptelinin and Nardi 2006) was entirely missed. For Bannon and Bødker, this ‘interesting’ linkage of artefact with mental conception was not “particularly enlightening” (2006) as it seemed to them too focussed on the artefact itself than the “artefact in use”. As Wilson (2006) and Walls (2009) observed, Bannon and Bødker seemed pre-occupied with ‘scoring points’ against their methodological opponents, which may have distracted them from conceiving any such synthesis.

Taken together with Wilson’s (2006) and Wall’s (2009) critiques of Kaptelinin and Nardi (2006), there are grounds to imagine that perhaps efforts towards any concrete operationalisation of AT in the systems analysis and design area (inclusive of CSCW and HCI) were at least partially hindered by such intellectual sparring across philosophical and methodological barriers.

2.7 Investigating AT’s Capabilities

Leading a concentrated burst of Australian AT research, Hasan (1998) suggested AT as a framework to supply what she deemed a lack in the IS discipline. Hasan identified at least three main areas of activities found in executive IS; use of the IS for decision support, the activities in creating the IS and the activities of those who maintain such a system. A crucial element for Hasan was the mediation of tools in AT. The identification of the information system with this tool (or at least some subset of these tools) was quite easy to make.

Deploying the deeper cognitive aspects of AT, Hasan pursued notions of exploring how data in a computer system can inform the generation of information and hopefully of knowledge. AT provided a useful framework for such investigations.

Whilst for Hasan, the IS was possibly itself an AT ‘tool’, and the creation and maintenance of the IS were probable ‘activities’, there was not yet any clear notion that AT could provide the basis for the design of the IS. The closest Hasan came to the notion was an oblique nod to Kaptelinin’s design checklists.

Importantly, Hasan, in reacting to Engström’s addition of groups, had not yet seen that AT could apply to a collective subject. The notion that there could be multiple activities within the group, with individual members as their subjects seemed elusive as these early AT thinkers seemed wedded to the notion that there could only be one activity under consideration at a time. Further, in locating an IS within an activity Hasan, though aware of the role of tools, located the system win the centre of an Engström triangle, possibly missing an opportunity to apply ATs more directly that had been seen before.

In taking an AT perspective to group work and organisations, Kuutti and Molin-Juustila (1998), appreciating that each community member is an actor (not just “cogs in the organisational machinery”) are able to conceive of a network of activities
which are connected. For Kuutti and Molin-Juustila, these connections take the form of rule generation, tool generation and, perhaps most importantly, material generation – whereby an outcome from one activity may provide the ‘raw material’ for the work done by another.

A few years later, Hasan (2001) had become more comfortable with the notion of multiple activities within a group structure, and had encountered calls such as those of Bodker or Nardi to deploy AT in the design of UIs.

Identifying HCI as the ‘cognitive’ and Computer-Supported Cooperative Work (CSCW) as the ‘social’ explorations of human mergers with information technology, Aboulafia (2001) conducted an authoritative comparison of the two and found several meta-theoretical problems. Aboulafia found that AT can provide a solution to these incompatibilities through its power to take a more abstract view of the whole of human ‘doings’. Drawing on notions close to the basis of SCOT, Aboulafia is intrigued by AT’s ability to accommodate the social construction of knowledge claims.

Concentrating on executive information systems (EISs), as had Hasan earlier, Gould and Verenikina (2001) found that an EIS without an interface aligned to business activities was of no effective use whatsoever to business operations. Their call was for AT to be deployed in the design of bespoke executive interfaces, tailored to the individual’s cognitive style. They were, sadly, silent on quite how this could be done.

To aid in BPM, and to compliment BPM tools, Ellison and McGrath (2001) proposed the System for Activity Theory Business Process Analysis (SATBPA). It was an important realisation that business documents and transactions could be viewed as the outcomes and components of activities. They also embraced Kuutti’s ‘collective subject’ notion, though this seems to have its greatest application when an entire organisation’s collective doings are somehow deemed to be one single activity.

Bødker et al. (2001) declared that computer applications constrain and/or discipline both use and design. AT provided a framework by which the fundamentally creative, or at least, less constrained, act of design might be conducted. For Bødker, this was largely accomplished through AT’s capacity to accommodate context. Bertelsen, in the same conversation (Bødker et al. (2001)) referring to the HCI notion of the information space, gave AT credit as a powerful tool to understand the ‘shared information space’ found within organisations. Bødker, perhaps still seeing AT as something of an explanatory anthropological framework (rather than as a genuine design concept in and of itself), bemoaned the difficulty in reconciling AT’s analysis of a user using a new design prototype, and a designer’s simultaneous act of evaluating that user’s experience, as there would have been two distinctly different Internal Planes of Action (IPAs) to consider, each with a quite different objective.

Larkin (2001) hoped AT could contribute in his approach to BPM, designed to overcome losses in “corporate memory” by capturing deeply contextual snapshots in time. For Larkin, AT put the organisation, rather than its databases or computer hardware at the centre of any such analysis. Two years later, Larkin (2003) had gone so far as to consider the use of Engström’s 1987 community-centred work, but was no nearer to proposing any prescriptive framework for BPM.
McGrath (2003) was prepared to go beyond the work of Larkin, and propose an AT framework for the development of a BPM ‘methodology’. Perhaps not surprisingly, McGrath reported there was little if any evidence of any prescriptive guidelines or tools for the practical application of AT. Following his earlier SATBPA work, he found a use for AT’s notions of motives in resolving the different understandings which different actors in a business had of their business doings.

McGrath proposed some fascinating uses for AT notions. He declared an entity called ‘artefact’ (a ‘thing’ exchanged between different activities in a network of activities) which was a curious merger of ‘tool’ and ‘object’ (facilitators of action with reason for action), a notion curiously reminiscent of the principles espoused in SCOT literature by writers such as Pinch (1986). The ‘understandings’ which McGrath sought to explain were tentatively given as some conflation of the ‘rules’ governing community behaviour and the ‘divisions of labour’ within that community; the two in combination somehow encapsulating a community member’s position.

By 2003 Hasan (2003) was reconciled with communities as activity systems, having finally broken the mould of the monolithic activity conception, and could draw parallels between the exchange of activity outcomes and the creation of knowledge in distinct contexts “Ba’s” as portrayed by Nonaka and Konno (1998).

Declaring a practical application of AT (missing in their 2001 work), Verenikina and Gould (2003) deployed Kaptelinin-esque checklists and scenarios, together with Vygotsky’s notion of the Zone of Proximal Development (ZPD). Verenikina and Gould see AT as the conceptual and philosophical common ground between these fragmentary proto-techniques, but had yet to declare any truly practicality for AT.

2.8 Putting AT to Work

AT found perhaps its first commonly acknowledged recognition by the broad field of computerised information systems, in the domain of Human Computer Interactions (HCI). The usability of systems via the agency their designed user interfaces is a topic of considerable academic study.

In the early 1990’s AT became a topic of interest to some in the HCI field, largely for its potential to describe and possibly explain the doings of a user. AT might, thus, help to uncover what Activities a user might require a system’s interface to support; help to explain the use made of a system’s interface; and/or to evaluate how successfully usable a system’s interface is.

Some other AT-flavoured work, particularly into the 2000’s, more directly targeted the elicitation of requirements or even the design of information and computer systems. It was necessary to survey these preceding methods, frameworks and approaches.

2.8.1 Bødker

Bødker’s landmark PhD thesis and text “Through the Interface” (Bødker 1991) and later, Nardi’s text “Context and Consciousness: AT and Human-Computer Interaction” (Nardi 1995) laid out AT as a useful tool and theoretical framework for
Chapter 2 – Literature Review

HCI study. Several proposals have come to light, notably the checklist idea (Kaptelinin 1999), however it has been stated that HCI has yet to benefit directly from AT (Vrazalic 2004).

2.8.2 Kuuti

Perhaps unsurprisingly, whilst researching Computer-Supported Cooperative Work CSCW, Kuutti and Arvonen (1992) identified considerable potential for the principles of AT to highlight opportunities to deploy CSCW systems. By treating each case as a single activity, but re-populating a new Engeström diagram for each of several crucial stakeholders, Kuutti and Arvonen could observe the interplay of contradictions between these different contexts. Through the application of some heuristics, they were able to indicate opportunities where some application of CSCW might have an ameliorating effect. There was no notion in that work as to how the CSCW system might be designed, nor that AT might inform such a design process.

Later, Kuutti (1995) questioned the theoretical bases under which HCI research had been conducted. A disparity between predictions from research and real world experience highlighted shortcomings of ‘traditional’ information processing cognitive psychology approaches. Kuutti advocated strongly the adoption of the principles of AT to inform new modes of research into HCI. Kuutti highlighted AT’s applicability across different levels of abstraction, its coverage of social context and its ability to address culturally-historically dynamic development of system use. Again, Kuutti offered no proposal for the use of AT as an informing theory to design either user interfaces or the systems they would relate to.

If a theory could explain a thing, perhaps it could describe a thing – further, perhaps it could describe how a thing ought to be. It would fall to others to make this connection.

2.8.3 Fjuk et al.

Still concentrating in the field of CSCW, Fjuk, Nurminen and Smørdal (1997) investigated the notion of articulation, whereby different stakeholders’ dynamically changing contexts alter their interactions. Articulations from stakeholders, it was argued, would be a powerful tool for understanding the rich interactions in CSCW environments. A theoretical basis was required, however, to interpret these articulations. To this end, they proposed integration of both AT and of Strauss’ theory on action. AT was seen to provide the ability to view articulations in the communal context.

Fjuk et al. (1997) expressed considerable regard for the potential of AT. They found it unfortunate that “activity theory [did] not provide us with a well-defined concept of collective action.” Once again, whilst AT’s potential applicability to understanding how people and technology might function together (as subject, community and mediating toolset) was observed, no attempt was made to ask if an explanatory theory might yield requirements for better tools.

2.8.4 Kaptelinin et al.

The earliest attempt uncovered in this literature survey towards using AT in a design-oriented sense was that of Kaptelinin, Nardi and Macauklay (1999). They proposed
a series of checklists whereby contextual issues impacting on the use of computers in real world scenarios might be examined, highlighting factors for designers to consider.

The method addressed either end of a systems development lifecycle, its design and its evaluation. By use of rather jargon-heavy and purely textual tabular checklists (informed largely by AT notions of mediation), Kaptelinin et al. strove to identify criteria for success and failure.

Falling well short of being a comprehensive systems analysis and design method *per se*, the Activity Checklist did highlight a direct role in design for AT. Further, the checklist’s notion could be seen as a precursor to the Vrazalic (2004) DUEM method for generating AT-flavoured usability evaluation criteria.

### 2.8.5 Gifford and Enyedy

While investigating Computer Supported Collaborative Learning (CSCL), and quite possibly having encountered AT through the cognitive psychology and pedagogy domains, Gifford and Enyedy (1999) surveyed the theoretical underpinnings of several computer-mediated learning environments and came to suggest the use of AT combined with distributed cognition. They called this new approach Activity Centred Design (ACD).

For Gifford and Enyedy, the goals of CSCL conflicted with the (largely cognitive) theories then in use. They observe a progression of centrisms informing computer-mediated education; from domain centrism, through learner centrism and then to their own activity centrism. These might be thought of as not incompatible with the progression of product, process and goal centric design, through user centric design to activity centred design suggested earlier in this chapter.

Gifford and Enyedy’s ACD focussed on three of the tenets of AT: mediation of activity through cultural artefacts, the need to recognise levels (in the manner of Leon’tev) and each internal activities was first instantiated in a social context. They recognise that under AT a mediating tool does not automatically *improve* a task, but certainly impacts upon it. The notion is suggested that, with an AT perspective, the design of a tool which *could* improve a task might be identifiable. The notion of ACD would be later reflected by Norman (2006) as superior to that of Human Centred Design (HCD) which he critiqued as insufficiently aware of humans in a social context.

Though a valiant and ground breaking step, ACD did not produce a widely applicable and prescriptive systems analysis and design method. Not only was it firmly rooted in the specific sub-domain of computer assisted learning, but Gifford and Enyedy (1999) had to concede it was “not yet clear how to apply the insights of AT to […] design rather than mere[…] critique.”

### 2.8.6 Jonassen and Rohrer-Murphy

Still in the field of computer aided education, specifically Constructive Learning Environments (CLEs), Jonassen and Rohrer-Murphy (1999) proposed a six-step
design framework which Quek and Shah (2004) critiqued as repetitive and confusing as to granularity.

Jonassen and Rohner-Murphy constructed a large series of deeply interrelated questions and cross-checks to guide the designer through their framework. Starting with an orientation towards the objects (motives) of activities, the designer could identify a learner’s (users) goals and thus identify Activities. An Engström styled examination of each activity would lead to decomposition into Actions and operations (after the manner of Leon’tev). The designer then (somehow) elicits tools and finally considers deeper issues of communal deployment and constraining rules.

The framework took an important step further than Gifford and Enyedy (1999) in that some notion of a prescriptive method emerged, firmly rooted in the precepts of AT. It was articulated that through AT, one could arrive at the required nature of some mediating tool to assist a network of activities.

Quek and Shah (2004) observe that the framework paid little attention to the notion of requirements elicitation, as the tools somehow became elicitable after a sufficiently AT-styled investigation of the learner’s (user’s) needs and contexts. Unfortunately the framework was only tested experimentally and there is little indication that it survived long enough in that form to be applied.

### 2.8.7 Korpela

Korpela, having suggested in 1997 that an AT method could inform the work of information systems professionals and could enhance the research of information systems later claimed (with Soriyan and Olufokunba in 2000), that the prime use of their Activity Analysis and Design (ActAD) method was the development of information systems. In this, Korpela at.al were perhaps the first to step beyond the somewhat niche domains of HCI, collaborative work and learning, to declare applicability of AT for information systems in general.

Their ActAD ‘day-to-day methodology’ is firmly rooted in networks of activities, though they consider the contributions of peripheral activities to some central activity (though they make it clear this restriction is for simplicity in presenting ActAD). They also recognise the need to temporarily set aside the historical aspects of AT to take a ‘snapshot’ for the purposes of design.

For Korpela et al., an activity may be visualised as a near linear procession from community, through (communicative) rules to various subjects, thence via tools (called Instruments) to produce in parallel both an object and an outcome. Of course in this, Korpela reflect the deeply fundamental AT notion that the outcome and Object might be compared. They offered a series of elicitation questions for the subjects, about what would be produced, what raw materials are required, what tools are used, social rules and final outcomes.

Quek and Shah (2004) pointed out a lack of direction in ActAD for the identification or production of new tools (the information system being developed) and a lack of notation and documentation. Showing many strong elements and powerful notions, ActAD remained in need of considerable further development.
2.8.8 Mwanza

Mwanza (now Mwanza-Simwami) in her 2002 PhD thesis presented a coherent six step Activity Theoretic method for HCI ‘research and practice’. This Activity-Oriented Design Method (AODM) was quite highly regarded by Quek and Shah (2004), though they observe its specificity in the HCI domain and its prominent reliance on mediation as evidence for ‘differing degrees’ of AT orientation.

A significant accolade for AODM was that, as of 2004, it was the only AT based information development method known to have been validated against real world application. Over two years two industrial test cases were used to validate the notions of AODM. It was noted however, that AODM specifically targeted a system’s HCI, and provided ‘guidance’ (clarifying the elicitation of requirements) rather than actual design. It could be argued that, in the literature, there was little evidence of AODM having been validated through the construction of any systems built to specifications declared through its processes. AODM remained however, one of the most validated of the activity oriented methods or frameworks.

The six steps of AODM consisted firstly of examining the ‘situation’, analogous perhaps to a problem-space, but more correctly the stakeholders’ socio-technical context. This was achieved by way of asking eight questions called by Mwanza ‘steps’ (sometime leading to confusions in the literature and called the ‘eight’ steps of AODM which, properly, actually has six steps). The eight ‘step’ questions directly elicit the activity itself and each of the seven nodes of an Engström matrix. It is unclear quite how the stakeholder is able to identify an activity, an achievement found difficult by even those well versed in AT.

In the second step, AODM produces an activity system from the answers to the eight elicitation questions. It was clarified that this was not a system of activities (plural) but rather a system of sub triangular relationships found within the Engström matrix. Despite the single activity, Mwanza specifically elicited subjects (plural) in the third of her eight ‘step’ questions. In this, it seems Mwanza stepped slightly outside of the precepts of AT and offered an expanded notion of the activity.

The third step of AODM decomposed the Activity ‘system’ (in reality, a singular activity) by consideration of ‘sub-activities’ identified by specific sub-triangles. The identification and deployment of these sub-triangles was Mwanza’s chief contribution and innovation to the pursuit of practicable application of AT to the computing field. Mwanza’s notion of sub-triangles appears below in Figure 2.16. The decomposition was largely performed via questions with the aid of a few heuristics (which primarily transformed AT jargon into terms more likely to be grasped by those that were interviewed). It was interesting that Mwanza preferred to decompose by way of identifying components of the activity, rather than actions or operations (as suggested by Leon’tev).

In its fourth step, AODM generated ‘research questions’ which more deeply investigated the transactions, tensions and influences within the six sub-trangles of the activity system produced by AODM. The answers to these questions lead into the fifth step, which directed observational studies. These would be particularly valuable
for the understanding of an information system’s development, rather than for developing it.

The sixth and final step of AODM interprets the findings of investigations by concentrating on the contradictions between and within activity systems. These contradictions (described in an earlier section of this literature review) were described by Engström (1987) but highlighted more by Kuutti (1996). It seemed apparent that the Mwanzian notion of sub-triangles may have emerged from an attempt on her part to locate and isolate Kuuttian contradictions. It is interesting that for Mwanza, contradictions may exist between sub-triangles, whilst for Kuutti they seem to have existed between a central ‘context setting’ activity and peripheral ‘provider/consumer’ activities.

![Figure 2.16: AT Sub-Triangles (from Mwanza (2001))](image)

It is unclear quite how the identification of problem areas, obviously of great value to consider, would directly produce, derive or even drive, the specification of a design for any information system. Any such system most certainly could have benefitted from consideration of problem areas, but AODM seems resolutely silent on offering prescriptive resolutions or guidance on their solutions.

61
Whilst many of the activity oriented methods described here have been little mentioned in recent years, AODM was revived as an evaluative tool to investigate mobile learning environments – a use for which it seemed especially well suited (Mwanza-Simwami 2009).

Mwanza’s contribution cannot be understated, as she took significant steps towards prescribing steps towards the analysis and design of systems. It did not constitute a self-contained SAD however and even Mwanza herself bemoaned the lack of pragmatic notions for operationalising AT in HCI (let alone beyond HCI).

2.8.9 Fjeld et al.

Fjeld, Lauche, Bichsel, Voorhorst, Krueger and Rauterberg (2002) described in detail the BUILD-IT tool, which mediated group work. An intensely visual multi-user product which made powerful use of then cutting edge virtual reality and gestural interfaces, BUILD-IT permitted group participation in a manner carefully engineered to comply with the principles of AT.

Fjeld et al. presented a fascinating example of a deeply AT-informed bespoke mediator, but did not propose any generalisation of how others might design other systems under AT. Indeed, there is no clear evidence that AT drove the design of BUILD-IT, but rather motivated and informed those who constructed it.

Providing no further concrete material for this current study, Fjeld et al.’s BUILD-IT is commended to the reader for further investigation as a powerful example of how innovative design might emerge when a suitable and coherent theoretical base informs the designers.

2.8.10 Collins et al.

Collins, Shulka and Redmiles (2002) deployed AT and a hierarchy of mediating artefacts towards identification of tools for customer support engineers. As with many AT writers (such as Vrazalic) Collins et al. seemed curiously bound to considering only a single activity and its component actions and tasks, which they termed a ‘single activity system’.

They explored ‘knowledge authoring’ wherein support staff at Hewlett-Packard documented solutions to customer issues, as part of a growing knowledge base. Certainly a non-trivial set of doings, this ‘activity system’ proved a challenge to describe in AT terms (following Engström).

Unfortunately, though fascinating and detailed, the paper essentially reports the ethnographic mapping of a complex group ‘doing’ in AT terms. Even in that however, Collins et al. found themselves challenged. Though AT “provided valuable insights” they were obliged to deploy complimentary models and concepts when communicating with others. Not unlike previous researchers, Collins et al. found that AT notions were difficult discuss (particularly with non-AT familiar stakeholders). Vygotsky’s classic dichotomy between object and outcome proved almost insurmountable for the subjects (stakeholders) in their study to grasp.

Collins et al. avoided burdening their subjects with the philosophical and psychological concepts of AT, in an attempt to minimise confusion. Unfortunately
they found subject confusion persisted. Perhaps there would have been value in deriving techniques and tools informed by AT, but set out in simple prescriptive steps.

When Collins et al. presented Engstrom matrix-based elicitation tools to their subject engineers, they struggled to populate the seven nodes. It is likely the case that AT serves best as a high level informing principle and not as a direct prescriptive toolset. There remains the challenge of creating simple and usable analytical tools (ahead of design processes) that are based on AT.

2.8.11 Vrazalic

McGrath and Uden (2000) observed that there is a near total lack of any prescriptive procedures for applying AT in software development. They found it difficult to apply Engström and Kuutti’s theoretical frameworks using prior AT case studies as a guide, as these were, whilst interesting, almost uniformly narrative in style lacking a well-defined or replicable process.

An exception to this lack of system and procedure was the work of Vrazalic (2001) who had grasped the power of AT to explore the political machinations within the development of information systems.

Later, Vrazalic (2004) proposed a method for evaluating the usability of a system after its completion. This technique was concerned with the broader social context in which the system is used. The user, her social environment, the system and all mediating technologies must be considered. Vrazalic adopts a broader distributed definition of usability in the manner of Spinuzzi (1999) that incorporates assorted genres, practices, uses and goals. Under this notion of distributed usability, Vrazalic considers the typical usability laboratory to be an artificial environment that has a number of shortcomings that can skew the results (Standish 1995). Vrazalic's Distributed Usability Evaluation Method (DUEM) deploys a comprehensive series of tests based upon Activity Scenarios generated from intensive observation of the user in their native work context, interviews with users and moderated focus group discussions.

DUEM consists of three interacting phases: understanding user activities; evaluating the role of the system in user activities and analysing and interpreting the results. The first phase produces a shared understanding of user tasks and goals. The second phase produces rich qualitative descriptions of the users’ interaction with the system. The third phase concentrates on identifying points of breakdown, where the system and the activity map contradict. The problems are described via deeply contextual definitions which aid in reaching any negotiated solutions. DUEM uses the notion of distributed usability and AT principles to define contexts of a system’s use by humans. Evaluation is adjudged against criteria derived from these initial findings, based upon user activity rather than system specific requirements. Users are deeply involved in an iterative process through interviews, workshops and observations. (Vrazalic 2004).

It has been observed (Brown 2004), and admitted (Vrazalic 2004), that one drawback of DUEM is that evaluators must have an understanding of AT principles to inform
their analysis and to help them guide users through the process. This precludes deployment by most software analysts and requirements engineers.

Vrazalic (2004) proposes in her PhD thesis a method for evaluating the usability of a system after its completion. This rich technique is informed by AT and thus is concerned with the broader social context in which the system is used. The user, her social environment, the system and all mediating technologies must be considered. Vrazalic adopts a wider definition of usability in the manner of Spinuzzi (1999) that incorporates assorted genres, practices, uses and goals. Under this notion of distributed usability, Vrazalic considers the typical usability laboratory to be an artificial environment that has a number of shortcomings that can skew the results (Vrazalic & Gould 2001). DUEM thus involves an involved and comprehensive series of tests based upon activity scenarios that are generated from intensive observation of the user in their native work context, interviews with users and moderated focus group discussions.

DUEM consists of three interacting phases: Firstly, understanding user activities, then evaluating the role of the system in user activities, and finally analysing and interpreting the results. The first phase produces a shared understanding of user tasks and goals. The second phase produces rich qualitative descriptions of the users’ interaction with the system. The third phase concentrates on identifying points of breakdown, where the system and the activity map contradict. The problems identified have a deeply contextual definition that aids in any negotiated solutions. DUEM uses the notion of distributed usability and AT principles to define contexts of a system’s use by humans. Evaluation is adjudged against criteria derived from these initial findings; based upon user activity rather than system specific requirements. Users are deeply involved in an iterative process through interviews, workshops and observations. Evaluators must have an understanding of AT principles to inform their analysis and to help them guide users through the process. (Vrazalic 2004)

The evaluations that result from DUEM show considerable promise however the wide scale deployment of the method may be inhibited by several factors; namely that the method requires trained evaluators, occupies a great deal of time (and is thus quite expensive) and requires that intended users be available at the time (Vrazalic and Hyland 2002). We must also consider that any usability evaluation method that is conducted after the fact can at best only indicate the quality or otherwise of the finished product or facilitate a late stage repair. It is questionable that any meaningful savings in the software production lifecycle would result, as each test is case-specific. If such an evaluation method were widely used, we might expect to see a slow generational or evolutionary improvement in the usability of systems, as each new product is evaluated and its success measured.

Previous publications by this author proposed that remediation of the software production process may be best achieved by modifying the design-side of the process. The agile approach of eXtreme Programming (XP) was strongly based on the notion of writing test cases well before the writing of code (Ambler 2009), unfortunately this notion existed in the systems design world and had seemingly little impact upon the considerations of the HCI community.
DUEM offers some crucial elements to be addressed in this design process; any resulting method would not only benefit from the principles of DUEM, but make DUEM itself easier and cheaper to apply. DUEM is based on user-system interactions rather than system requirements, because these requirements do not address such issues. If they did however, DUEM need not be such an end-heavy process.

### 2.8.12 Martins

Martins and Daltrini (1999) proposed a three stage framework for eliciting system requirements using AT. Step one asked for the declaration of activities. Step two identified Engström styled activity elements and step three decomposed into Actions and operations. The framework provided tabular columns for the listing of activities, then of actions and of operations after the hierarchy proposed by Leon’tev. Quek and Shah (2004) critiqued the framework for providing little support for the identification of Activities, providing as it did little more than a table for them to be listed into.

Martins (2007) gave a more detailed and developed description of the framework, with procedures given for each of the three stages. A particularly interesting first process in step one was to identify candidate activities, which are found through some application of common sense and experience in what he called a ‘kind of brainstorm’. It is important to see that Martins accepted the vagaries of appeals to common sense by acknowledging these ‘first cut’ activities might not persist in the face of further analysis.

Martins (2007) could confirm the presence of a genuine activity from the initial list of candidates, through the ‘principles’ of activity, action and operation (as described by Leon’tev). Each of the principles Martins described directly reflected the precepts of AT; for example, an action must be driven by a goal that supports the context of the larger activity. Martins’ framework does not provide specific tools, techniques or heuristics for these ‘confirmation’ steps. Quek and Shah (2004) find the lack of guidance of these steps (as they were in 1999) a problem, and this author found no hard evidence in the 2007 paper that this shortfall had been adequately addressed.

Though admirably committed to the core precepts of AT, Martins framework presented little more than a notion of how requirements might be elicitable from a consideration of activities, their social-historical contexts and their hierarchic decompositions into actions and operations, without explaining how this might be achieved. Well ahead of its time in 1999, the framework had made insufficient progress by 2007 to have become a readily deployable, prescriptive or usable method.

### 2.8.13 Uden et al.

Uden, Valderas and Pastor (Uden et al. 2008) proposed an activity theoretic model for analysing the requirements of web applications. The context of the work, being Web applications, implicitly carries the notion of system design as, for Web applications, the interface (the web pages and their interface elements, dialogues and controls) comprise perhaps a greater proportion of the overall system than for some other classes of system. In terms of this author’s prior work, such applications have a High Interaction Quotient (High-IQ) (Brown 2005).
Uden was careful to outline three crucial limitations of AT for her work; firstly that the researcher must spend time to acquire a complete understanding of the activity system under observation (though, once acquired, such understanding is of great value). Secondly, the decoding and unravelling of the organic complexity of any human activity system is difficult. Finally, Uden clearly highlights the inherent problem that arises from Leon’tev’s layers, the difficulty in differentiating and classifying observed doings as activities, actions or operations. “… finally, there is the difficulty of distinguishing between the levels of activity, action and operations.” (Uden et al. 2008). For Uden however, the benefits delivered by the deployment of AT as a ‘broad theoretical framework’ exceeded these limitations.

Uden applies the work of Bardram (1998), to assist in coping with Leon’tev’s layers; whereby a condition (the driver of an operation) is unaware of a common objective, a goal (driver of an action) arises when a common object has become stabilised but, interestingly, an object proper (driver of an activity) forms only when there is some blockage and cooperative work is required towards resolution.

Uden et al. (2008) particularly highlighted the cultural-historical dimensions of AT. For AT, practice is reshaped as time passes, form is a product of history. Uden et al., focussing on Web applications, maintained that a single laboratory experiment could not capture “how tools are used”. It was important for Uden to highlight and accommodate a variety of usage styles and she deployed layers of history in an attempt to capture these details. The Web, of course, might be seen as a consumer-end interface, quite literally broadcast to an unknown and unknowable suite of possible users, often in the hope of enticing their interaction. It is vital for the commercial success of Web applications, designed to attract and retain customers, that the design not feel tightly constrained or prescribed. Alternately, for a business organisation, wherein practice is somewhat prescribed by training, directive and adherence to motives arising from stated business goals; where usage is less invited and solicited than dictated and reimbursed, it may be the case that cultural-historical evolution and development of usage is less of a factor. Perhaps the AT investigation of how business organisation members use a system might best be viewed with an historical snapshot. The lesson however is taken, that even in a highly prescriptive business context, cultural-historical factors could influence the activity system. Issues of re-factoring and re-engineering for reuse across temporal change must not be ignored, even though such changes in the structured business context might be considered to occur more slowly.

Not uncommonly for AT flavoured work, the paper proposed a set of elicitation questions and considerations. Curiously however, there were aimed at completing action-level descriptive ‘shells’, each naming the action, stating its goal, listing its subjects and recording any further pertinent data. Whilst Engström clearly presented a matrix of seven nodes at the Activity level (the basis for all AT analysis), it seems Uden et al. were satisfied with an action level ‘matrix’ which aside from action name contained just two ‘nodes’ (one of which was the Activity level notion “subject”), plus a comment field for ‘everything else’ including constraints upon functional requirements (such as cycles per hour of a repeated action).

Innovatively, Uden et al. declared two types of operation, to represent those doings performed within the system, and those involving the user. This distinction is
analogous to the declaration of a system automation boundary (as found in many
flavours of the traditional ‘structured’ systems analysis and design method. To assist
in distinguishing and documenting the two classes of operation found under her
method, Uden deploys ‘activity diagrams’ as described under UML to describe
operations in sequence. At this point Uden et al. were required to propose a protocol
for mapping terms from the UML activity diagram to AT.

It is interesting to note that whilst Uden et al. were cogniscent of difficulties in
navigating Leon’tev’s layers, and conceived of systems of related activities, the main
element of analysis they presented was at the action level, together with use of
hybridised UML diagramming tools at the operation level. It remained clear that the
breadth, scope and vast applicability of AT, especially including its cultural-
historical factors, needed to be pinned down in some manner to allow a prescriptive
declaration of system and/or interface requirements, though Uden et al. might not
agree that a traditionally declarative result is sufficiently sensitive to the ‘“softer’
people-related issues of software systems.’” (Uden et al. 2008)

The limitation imposed by the need for an analyst to obtain complete understanding
of an activity system was reminiscent of the limitations attributed to Vrazalic’s
DUEM method (Vrazalic 2004, Brown 2004). In both cases, the analyst seemed to
require a solid grasp of the precepts and principles of AT, and to have applied them
deeply, prior the commencement of the method proper (albeit one of analysis, design
or of evaluation). Such a limitation could have curtailed AT’s applicability for
neophyte analyst-designers. This highlighted the need to somehow strive build a
prescriptive method to assist neophyte analyst-designers, informed by some AT
principles but without necessarily requiring deep personal expertise in AT.

2.8.14 And Beyond

Quek and Shah (2004) reviewed five of the works surveyed above. Naming them as
‘Activity-Based methods for information systems development’ Quek and Shah
addressed the Korpela’s ActAD (1997), Kaptelinnin et al.’s Activity Checklist
(1999), Mwanza’s AODM (2001), Jonaseen and Rohrer-Murphy’s framework (1999)
work. They investigated the breadth of each method across the information
development lifecycle, questioned their coverage of AT concepts, and checked for
real world validation. On these factors, Quek and Shah found considerable scope for
further research and development as few offered convincing validation (only
Mwanza showing a real-world test at the time), none but Mwanza’s AODM covered
more than a small segment of the lifecycle, and the application of AT was non-
comprehensive (several methods showing a need to incorporate other theories or
methods). Though a leader on two of their score’s, Quek and Shah critiqued
Mwanza’s AODM for principally applying to HCI, and for applying AT in ‘differing
degrees’ given prominence to mediation over development.

Applying Leon’tev, Norman (2005) differentiates between tasks and activities,
making an activity an integrated and coordinated set of tasks. Whilst Norman
claimed adherence to his “own brand” of AT, its precepts were readily recognizable.
In discussion of design in a broad sense, inclusive of hardware devices as well as
software systems, Norman declared that application of activity centred design (ACD)
Chapter 2 – Literature Review

necessitated deep understanding of both people, but also of the tools they use and their motivations. This bespeaks a deeply classical understanding of AT. Norman went on to state quite categorically that, “successful devices are those that fit gracefully into the requirements of the underlying activity, supporting them in a manner understandable by people. Understand the activity, and the device is understandable.”

Later, Norman (2006) went further, and declared that good design, to human-centred principles, organised things taxonomically, by topic. He found that a source of failure for many ‘well-designed’ software, because as tasks are actually combined into activities, one might need both a hammer and a nail, only to find that the designer had neatly stored each in quite separate places. Norman saw a solution: “the correct approach to the support of behaviour is activity-based classification”; though he found a need for both activity-centred and human-centred design approaches.

Schweikardt (2009) found that pure user-centrism, though admirable in its ability to deliver what an end-user wants or needs, can lead to designs which poorly serve the broader community. Citing largely ecological concerns arising from user-based designs, Schweikardt considered the user focus as misguided. Norman’s ACD approach, with its broader activity context, showed some promise for Schweikardt, though he ultimately called for a balance between the use of resources to meet user needs and an economy of design for the sake of the society as a whole.

Williams (2009) offers a comparative review of user centric design (UCD), which arose through the work of Norman over 20 years; AT inspired ACD, and Cooper’s goal directed design (GDD) developed between 1983 and 2000. Williams chose to distinguish between ACD’s concentration on ‘what’ tasks are performed and GDD’s concentration on ‘why’. Williams overstates the case at times, particularly given that she acknowledged the pure AT notion that a computer tool (such as a web application) would become the tool that enabled a user to achieve a goal. Williams observed that ACD had “no authoritative text [...] intended for practitioners”, and showed a gentle preference for GDD’s structure and prescription. Unfortunately for GDD, Williams cites Goodwin’s (2009) assessment, that a skilled designer would need at least a year to master even the most basic elements of GDD. In overview, Williams presented no convincing argument against ACD, aside from its lack of any authoritative or prescriptive methods. Indeed, if an AT method were available that attended to goals, to user needs and could be instructed in less than one year of intensive training, one might suppose Williams would have declared it the ‘winner’ of the review.

AT has shown great promise in the literature, in HCI, highly interactive systems such as Web Applications (which serve as a sort of bridging domain between HCI and more general SAD) and even beyond. It even came to have its own acronym “ACD”. Unfortunately, with no clear prescriptive methods or techniques available in the literature, it remained an admirable informing principle, a notion, something to inform what Norman (2005) would call the ‘attitude’ or ‘mindset’ of the designer.
2.9 Findings: Challenges for SAD and HCI

In under a half century, the software and information systems domains (inclusive of their sibling disciplines of usability and human computer interaction (HCI)) have suffered from a profusion of methods both formal and *ad hoc*, yet have consistently suffered a widely acknowledged (if little discussed) unacceptably low success rate.

The historical review of the literature presented above demonstrates that methods for guiding the lifecycle of a software project (or elements thereof) which were frequently created to address perceived issues of quality and user acceptance, often did not long survive introduction with practice. Taxing overheads of training, or of a method’s inherent bureaucracy, combined with a lack of any formally recognisable engineering rigour (in the manner of the traditional professional bodies which accredit civil, electrical or mechanical engineers), permit *ad hoc* adoption and hybridisation at the commercial coal face. Misapplication of methods only adds to any flaws which may exist in those methods themselves.

The 1980’s and 1990’s saw an increase in the number of methods, ‘methodologies’, techniques and tools available to the practitioner and theoretician alike. This author has been unable to locate any evidence (primary or secondary) indicating any significant lift in software project success rates as a result, indeed the historical literature review given above consistently describes a less than glowing picture. What did result was a deeper fracturing of the professions within this mosaic domain of practice. Gentle ‘wars’ of philosophy were visible in the literature and established lines of miscommunication and segregation were visible in practice (Draper and Norman 1984, Kaptelinin and Nardi 2006, Wilson 2006, Walls 2009). Entire sibling domains, such as HCI and SAD, tended to avoid one another. A result has been that the customers and other stakeholder (such as end users) have been the subject of vigorous debate. A widely held perception that the user had been forgotten or underrated underscores the rhetoric in many methods and approaches. The question of how best to ‘handle’ users’ famously volatile requests is still with us. The HCI community, in some regards, laid a claim to championing the user, and make it a goal to apply as best a ‘face’ as possible over the user-unfriendly code produced by those in the wider SAD community (Faulkner and Culwin 2000). The notion that the user interface is an inseparable element of the system itself, being as it represents that part of the systems defining border (or *skin*) which users *sees*, has had little airplay.

One branch of practice bravely struggled on using increasingly disjointed components of the aged traditional structured methods, overlaid with occasional elements drawn from newer methods. The object oriented approach, which grew quietly from the 1960’s before its explosive recognition in the late 1980’s, is essentially the coder’s response to a chaotic world – establishing a justification for building systems in a manner which suits the builder. The increasing perception of customer dissatisfaction with delay, dysfunction and poor usability resulted in the near heretical breaking of the traditional method – into spirals, evolutionary processes and then into horizontal or vertical notions of incremental delivery, often of prototype product. These were *sufficient* to ‘keep the customer happy for now’ and, as might be expected, have been criticised as stop-gap measures. A criticism of
Chapter 2 – Literature Review

prototyping and incremental deliver was that the customer served as an unwitting ‘beta-test guinea pig’.

The 2001 Agile Manifesto (Beck et al. 2001) codified a set of principles and relative valuations to inform an AGILE methodology (or approach) but individual methods under that approach were quite distinct in scope, technique and effect. Criticism has been that the manifesto produces a licence to hack, though this is certainly an overstatement and an unrealistic portrayal of the methodology. It may be true however, that at the deepest levels, the notion of quickly addressing the ever-shifting customer requirements was driven by a desire to ‘sufficiently satisfy them and get paid before they change again’. The rapid development notion codified this into a series of discrete vertical increments, each delivering a contracted project deliverable. Each new component launched a freshly negotiated micro-project. This serves more as a method of customer management, or contract control than as an innovative SAD method per se.

What has been seen through this review of the literature, is that methods tend not to survive contact intact with coal-face practice for too long. Procedural overheads, such as that of classical structured design in its early form, and extended training demands (unacceptable in a commercial environment) such as that for the goal directed design approach, tend to see methods breaking up into disjoint chunks of apparently usable practice. Ad hoc and hybridised suites of practice (often undeserving of the title ‘method’, much less that of ‘methodology’) grow organically as generations of analysts and designers evolve techniques in communities of practice (Wenger 1998) and then pass their skills on (often through supervision, coaching, mentoring and tacit transfer) to their successors.

To ease acceptance and usefulness of any new approach, a challenge for this research then is to strive for new methods to be agnostic of pre-existing methods and not unfriendly towards as many implementation technologies as possible.

Neophyte entry to the workforce faced something of a glass barrier, as practice with hybridised ad hoc toolsets often relied on intuitive leaps and bounds, driven from an analyst-designer’s prior experience. The practice of SAD remained an art, conducted by master artisans, as attempts to engineer it failed in sequence to change the culture. Small wonder then, that it became difficult to identify coherent underlying methodological underpinnings, or to observe coherent and efficient communications between phases of a software projects lifecycle.

As seen in the DUEM method for evaluating user interfaces (Vrazalic 2004), too high an analytical overhead at the end of the project lifecycle when usefulness of the result might be reduced, and too high a reliance on analyst expertise reduced applicability and hindered uptake. This excellent method remains, so far as the literature shows to date, on the academic shelf. Other literature in the smaller lifecycle side-band of HCI and usability shows a clear call for just such an informing theory to reunify its components. A similar challenge remains sui generis for all of SAD. A clear hope is that an informing principle could permit better communications between elements (phases, sections, persons) within a systems analysis and design (if not across a larger portion of a software projects’ lifecycle), and perhaps between sidebands, long sundered by near impenetrable paradigmatic norms.
Communication issues are a factor cited in some reports on failure rates. One objective for this research then is to shift the analysis of AT criteria (perhaps in a manner inspired by DUEM) to the analysis and design phases nearer the front of a software project’s lifecycle, and incorporate the theory into simple design steps to permit ease of uptake and use.

The literature reveals that ‘poor requirements’ are a source of low success rates. Some requirements are poorly implemented or described (an issue related to the lack of a single consistent informing theory) and others are perhaps poorly grasped in the first place. Extensive formal requirements elicitation techniques impose time delays which can frustrate analyst-designers and invite revision by dissatisfied stakeholders and end-users. Overly light or cursory requirements elicitation techniques can fail to capture what was needed and/or reach a mutual understanding with the stakeholder. Rigidly formal methods can fail to cope with so-called non-functional requirements and overly coder-friendly techniques and approaches can put the cart-before-the-horse, too often resulting in delivery of something other than that required. Agile methods run the risk of delivering hasty product, cobbled to meet immature requirements, perhaps justified by an assumption that the user simply cannot reach a stable conclusion for long.

Another objective for this research then, is to explore if, with deployment of an informing theory that addresses complex human ‘doings’ in some sufficiently non-burdening way, requirements could be more meaningfully captured with some greater confidence as to fidelity and consensual agreement.

The literature reveals a splintering of loyalties between approaches and methods, underlined by a somewhat reactionary movement towards abandonment of the quest for any single methodology. The erosive, if not explosive, reception awaiting methods as they hit the coal-face of practice generates any number of favourite method-fragments used in hybridised combinations by professionals and those they supervise and coach.

To summarise the challenges identified here:

- SAD and HCI methods alike tend to lack single, coherent informing philosophical or theoretical bases.
- SAD methods lack clear lines of communication between their phases (however they are arranged, or how tightly or loosely they are defined), to its side-bands such as HCI, and between analyst-designers and stakeholder-users.
- Requirement capture is poorly understood and poorly conducted. There are grounds to consider selecting an informing theory which readily grasps the complexities and scaling issues of user organisations.
- SAD methods tend to lose original focus or coherence and be adapted into expanded and/or ad hoc collections of practice because of high bureaucratic overheads, learnability and/or ease of use issues often resulting from conceptual complexity.
Current and ad hoc methods in practice rely too heavily on accumulated tacit knowledge and experience, making them difficult to transfer to new entrants and thus discouraging neophyte analyst-designers.

These challenges were (and are likely to remain) non-trivial, and perhaps even systemic and symptomatic of the domain *sui generis*. There was a significant risk that the one genuinely common factor across both time and modes of practice was the perceived low success rate. Unlike the natural world, where the experiencing of time, of gravity, of the daily and seasonal cycles and of life itself are independent of culture, language, profession or historical period; software inhabits in and of itself an entirely artificial environment, users and designers cannot share a deep biological and culturally point of view, for there is none to be shared. If success is a socially negotiated consensual phenomenon, as has been argued by Kuhn (1962) and others of scientific ‘truths’ themselves, then the lack of any common ground can only complicate and hinder consensus formation by the incommensurability of the participants’ viewpoints.

This research however, maintained a (perhaps unjustifiable) optimism, that if some appropriately broad informing theory could underpin a method, whose bureaucratic overhead was light, whose learnability was adequate, whose reliance on prior experience was low and whose exclusiveness (towards current tools, techniques and methods) was low; then perhaps there was grounds to attempt a beginning in building such a method.

It remained to define a structured and justifiable research methodology under which the challenges might be addressed.
Chapter 3  Research Methodology

3.1  Introduction
The review of the relevant literature in chapter 2 identifies a number of challenges for both the development of both software systems and their user interfaces. An unacceptably high rate of rejection, cost overrun and re-engineering appears to be at least partially attributable to poor specification of system requirements and insufficient consideration of interface usability. There appears to be little if any reliably systematic or prescribed methodology for transforming users’ needs into useful requirements that will inform construction of a usable and functional system within time and budget. This seems particularly true for high-IQ systems, which has been previously (Brown 2005) defined as those systems whose function and purpose are highly related to direct user interaction – and thus, whose functionality resides conceptually close to the system interface(s).

As noted, the existence of the Agile class of practices and the emergence of the Method Engineering concept, together with evidence of slow and low industrial uptake of methods (Kerkow et al. 2005), seem to indicate a crisis of faith in software engineering.

The discussion suggests that there is cause to move from User-Centrism to Use-Centrism, from an Agent or Actor orientation to an Activity orientation. The well known Activity Theory (AT) seems an acceptably strong candidate basis, especially as it has been an informing principle for a number of approaches to both System and User Interface (UI) design (detailed in chapter 2), even if none has as yet produced a prescriptive method.

The hypothesis emerging from the literature review in Chapter 2 was that:

Systems Analysis and Design (SAD) may be improved if conducted through a prescriptive but agnostic method, constructed according to Activity Theory principles.

The agenda of this thesis therefore, is to explore the suitability of AT as the basis for a new approach to analysing and designing systems and their user interfaces.

This chapter begins by clearly stating the goals of the study. It then explores a research methodology based on the framework of March and Smith (1995). This framework has been deployed successfully in prior studies of Design Science. The selection of this framework will be justifies by comparison against some other commonly used frameworks. March and Smith’s framework will serve as the abstract framework for the research methodology used in this study.

The second half of this chapter will explicate the research methodology in finer detail. As the goals of the study are to make improvements against the observed challenges, a Normative Research approach is applied. A normative implementation of Routio’s (2005) four stage research process is provided in detail.

The next section reviews the research goals of the current study.
3.2 Research Goals
The research study has four quite distinct and sequentially linked goals. These are:

1. To identify challenges in existing SAD methods.
2. To hypothesise some means of improvement to address the challenges.
3. To construct a SAD method, based on the hypothesis.
4. To evaluate the new method against the identified challenges, by applying it against a realistic case study.

Having set out the research goals, the following sections describe the selection of a suitable research methodology.

3.3 Choosing a Research Framework
Given the research goals identified above, in section 3.2, which revolve about the construction of a new SAD method, it is reasonable to adopt a research methodology suited to such an activity.

Vrazalik observed (1995) that research of this kind may be conducted under a framework designed for Information Systems (IS) research. There are numerous IS research frameworks available. There are a number of IS research frameworks for general research activity. Others are often closely tied to specific nodes of research activity such as; survey research (Pinsonneault & Kraemer 1993), ethnographic research (Myers 1999), case-study research (Gable 1994; Cavaye 1996; Darke et al 1998) or interpretivism (Klein & Myers 1999; Walsham 1995).

March and Smith's (1995) framework, based on the design science approach, would seem to be well suited to the current project which involves the construction and testing of a method. This framework serves to locate the study at the most abstract level.

At a finer resolution, since the goals identified in the previous section involve an attempt to improve upon some aspects of existing SAD methods, a normative research strategy will be deployed. A normative methodology, as described by Routio (2005) is well fitted to the construction of artefacts (including methods) and their evaluation against characteristics of preceding artefacts (and methods).

The following sections will detail March and Smith's research framework and the normative case study framework.

3.4 March and Smith's Research Framework
March and Smith adopt a Systems approach in their 1995 article “Design and natural science research on information technology”. For the authors, a system is where Information Technology (IT) is instantiated; IT being defined as “technology used to acquire and process information in support of human purposes” (March and Smith 1995). Given that the current study explores an activity-oriented, or use-centric conception of systems design, the approach described by March and Smith seems applicable for investigating the construction of a computer based system, designed to support purposeful human activities.
March and Smith observe a division in technology research. They describe two broad categories: descriptive and prescriptive, wherein descriptive research, not unlike natural science research, produces knowledge about the nature of technology. Prescriptive research however, which March and Smith link with Simon’s (1969) design science, deploys the results of descriptive research to improve the performance of technology. Whilst the authors consider descriptive research to be somewhat more classically and theoretically pure, they observe that prescriptive research has been more successful. To address both positions March and Smith propose a two-dimensional research framework, which is described below.

### 3.4.1 Theoretical Background

March and Smith’s two technology research categories; descriptive and prescriptive, may be reflected in the common divisions of basic and applied research. Basic research, as referred to by Järvinen (1999), is associated with natural science research, where classical scientific activity attempts to determine facts from observations of phenomena in the natural world. Kaplan (1964) describes two active phases in natural science practice: discovery and justification. Discovery constructs, proposes and claims descriptive models of natural phenomena under labels such as ‘hypothesis’, ‘theory’ or ‘law’ in the hope that these may explain some underlying reality about nature. Somewhat naively put, the goal of natural science research is to know more about the actual truth of universal reality. The issue is not so much the construction of an explanatory model which conforms to observation, but the validity and the fidelity of the model.

Justification attempts to substantiate these models by way of some paradigmatic and agreed procedure of testing, typically referred to as experimentation. The value, persistence and acceptance of these models and the traditional tests applied to them, are negotiated by a sufficient number of acknowledged experts from a given domain, against mutually agreed norms of ‘truth’ and the explanatory power of their testing procedures and analyses. As no human may have absolute certain knowledge of the sensory inputs of another, only consensual negotiation, based upon mutually agreed artificial paradigms of analysis and testing can resolve ‘truths’ about nature. Since the activity of natural science is to uncover the truth of the natural world, and the only absolute measure of success is the truth of the natural world, then natural science research can never operate with any surety. The only claims which can be made with confidence are those regarding rigour with which artificial procedures of natural science research practice were followed. Irrespective of their efficacy, these natural science research procedures may be considered to have four common components: theorising about a phenomenon, building some test, evaluating the test and justifying the product.

The actual nature of such natural science research activity is the subject of much debate and historiographical and methodological study. One commonly quoted position (Popper 1934 – English version 1959) declares that researchers should, and do in fact attempt to *falsify* their models during the justification phase, and constantly seek to find better explanatory models of natural reality. Others (Lakatos 1976) find this position logically admirable but cannot reconcile it with the history of scientific research. Other studies (Merton 1938, Kuhn 1962, Schuster & Yeo 1986) seek to uncover the sociological and communal aspects of scientific practice.
Chapter 3 – Research Methodology

The only reliable fact to emerge from any serious study of natural science research is that the models it ‘discovers’ and ‘justifies’ can never be considered the ‘last word’ on the truth of reality, as nature itself is resistant to absolute and unquestionable analysis. At least, humans seem quite unable to ever be sure of the ‘absolute truth’ of their models of natural reality and their test findings.

Applied research however, not being concerned with the reality of natural phenomena, is therefore concerned with human artefacts. Paradoxically, applied research bases its work on the ‘findings’ of natural science research (Järvinen 1999). There is no claim to ‘discover’ or to explain however, but rather to construct and prescribe artefacts and procedures for the betterment of human activity. It does not attempt to justify itself against a fundamentally unknowable reality, but rather against the more subjective perception of human utility and value.

The conception and construction of artefacts may be considered an act of design, just as the result of the same conception and construction may be termed a design. The word design, being both noun and verb, therefore describes both process and product (Walls et al. 1992). Research that leads to the production (rather than ‘discovery’) of things useful to humanity has been described by Simon (1969) as Design Science and a similar meaning is conveyed by livari’s (1991) term, Construction Research. Interestingly the rigorous paradigmatic processes deployed by natural science research, in an attempt to circumvent human frailty, are themselves artefacts and as such, the result of design science or construction research, though some natural science researchers might dispute this.

The value and success of design science, or constructive, research activity need not be measured by questionable ‘tests against nature’ as with traditional natural science research. The artefacts resulting from design science need only be tested against utility, which is just as negotiative and subjective as natural scientific debate, but is at least free from the rhetorical and philosophical difficulties of discussing the ‘truth about universal reality’. The value of design science artefacts may be made with greater surety. If a sufficient number of stakeholders and domain experts reach a sufficiently consensual agreement on the value or utility of an artefact then it is literally more useful as a human artefact. There is no need to consider if it is a ‘truer’ representation of some underlying universal reality. IS research, specifically, has value when its application improves practice (Galliers & Land 1987). This position is not incompatible with Reichgelt (2004), who calls for the establishment of an identifiably distinct IT research agenda which has, as its central position, the exploration of the values and costs of IT products.

3.4.2 Design Science Artefacts

According to March and Smith (1995), design science produces four hierarchically specific categories of artefact.

1. Constructs, which are the atomic conceptual terms used in the domain specific language that describes issues and problems within that domain. Constructs are conceptually similar to the literals of first order logic.

2. Models, are propositions expressing relationships between constructs. Models are conceptually similar to propositions in first order logic.
March and Smith (1995) are careful to observe however, that in design science models are more concerned with utility than truth. Therefore a greater degree of representational inaccuracy may be permissible with models than with first order logic propositions.

3. Methods, describe procedures for conducting goal-oriented tasks. They may be assembled from, or driven by, constructs, models or both. These constructs and models may be implied from the adoption of a method.

4. Instantiations are artefacts, comprised of the methods, models and constructs put into use. The utility and viability of constructs, models and methods may be observed in the operation of the instantiated artefacts they have informed. Nunamaker et al (1991) explores the interesting case where artefacts precede constructs models and methods. Under this premise, design science may seem to take on some aspects of the natural science approach and attempts to explain an artefact. Curiously, an artefact which precedes the articulation of formalised constructs, models or methods may not yet actually be an instantiation – or it may be considered an instantiation of constructs, models and methods yet to be uncovered or articulated. The philosophical implications, fascinating as they are, lie beyond the scope of this project.

Figure 3.1, represents the four artefact categories and their inter-relationships. Constructs can be seen to inform both models and methods. This is clearly necessary as constructs are the terms in which models and methods are written. Instantiations are derived from methods directly, and the operation of the instantiated artefact feeds back to the constructs, models and methods. In this way, instantiated artefacts can be both the conceptual test of the methods and their underlying constructs and models, and (if feedback is accepted and iteratively incorporated) a refinement and formalisation tool for them.

What sets design science apart from less directed artefact creation, are the goals of innovation and utility. The objective is improvement. Design science therefore must both assemble and evaluate artefacts. An artefact is assembled, or built, to a specific purpose. March and Smith (1995) consider progress to occur when design science can replace an artefact with more effective ones. Evaluation is required therefore to establish if progress actually occurs. Even if little or no progress occurs, which is to say, the artefact functions but not markedly better than its predecessor, then the underlying constructs, models and methods have still been validated in so far as an instantiated artefact could demonstrably be built.

While conceptually separable, the building and evaluation of instantiated artefacts are interdependently tied to the environmental and human contexts under which they are deployed. Recalling the four abstract components of natural science research, namely theorising, test building, evaluation and justification, March and Smith (1995) propose a two dimensional research framework which encapsulates both natural science and design science by cross-referencing these four research activities against the four categories of output; constructs, models, methods and instantiations.
3.4.3 Two Dimensional Research Framework

A two dimensional sixteen cell matrix, as shown in Table 3.1, is used by March and Smith (1995) to represent what they term ‘viable research efforts’ (p. 260). A research effort may occupy one or more of these cells, each with its own research procedures and objectives.

<table>
<thead>
<tr>
<th>RESEARCH ACTIVITIES</th>
<th>Build</th>
<th>Evaluate</th>
<th>Theorise</th>
<th>Justify</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Method</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instantiation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.1: Two-dimensional research framework (from March and Smith (1995))

Since artefacts are constructed to serve a specific purpose, March and Smith (1995) indicate that research efforts located in the ‘build’ column of their framework need to be evaluated according to their usefulness to some specific group of users. If the artefact being built is a replacement for some pre-existing artefact, and thus constitutes redevelopment, then the contribution or significance of the research effort is measured by the degree of improvement made over the preceding artefact. If the artefact is a new development then the contribution or significance of the research effort is indicated by the novelty of the artefact. If the artefact utilises innovative constructs and models, yielding innovative methods, then these are justified and validated in the degree to which an instantiated artefact can function.
It is possible to conduct a research effort which is both a redevelopment and a conceptual innovation. Such an effort may deploy innovative constructs and models and/or deploy constructs and models in an innovative way. If the method and its instantiated artefact are functional, then there is value in the conceptual innovation. Should the instantiated artefact exceed the performance of extant artefacts in some key indicators then there is further contribution by way of its redevelopment value.

In the case of redevelopment contributions, the ‘evaluation’ column of March and Smith’s research framework requires the construction of metrics and criteria by which the improved performance may be gauged. These criteria must correspond with the outcomes the artefact is designed to achieve.

March and Smith (1995) offer a set of metrics for consideration in each of the four research outputs of their framework. Table 3.2 lists each of these metrics and criteria against the appropriate research output.

<table>
<thead>
<tr>
<th>ARTEFACT</th>
<th>METRICS/Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructs</td>
<td>Completeness</td>
</tr>
<tr>
<td></td>
<td>Simplicity</td>
</tr>
<tr>
<td></td>
<td>Elegance</td>
</tr>
<tr>
<td></td>
<td>Understandability</td>
</tr>
<tr>
<td></td>
<td>Ease of use</td>
</tr>
<tr>
<td>Models</td>
<td>Fidelity with real world phenomena</td>
</tr>
<tr>
<td></td>
<td>Completeness</td>
</tr>
<tr>
<td></td>
<td>Level of detail</td>
</tr>
<tr>
<td></td>
<td>Robustness</td>
</tr>
<tr>
<td></td>
<td>Internal consistency</td>
</tr>
<tr>
<td>Methods</td>
<td>Operationability (the ability to perform the intended task or the ability of humans to effectively use the method)</td>
</tr>
<tr>
<td></td>
<td>Efficiency</td>
</tr>
<tr>
<td></td>
<td>Generality</td>
</tr>
<tr>
<td></td>
<td>Ease of use</td>
</tr>
<tr>
<td>Instantiations</td>
<td>Efficiency</td>
</tr>
<tr>
<td></td>
<td>Effectiveness</td>
</tr>
<tr>
<td></td>
<td>Impact on environment</td>
</tr>
<tr>
<td></td>
<td>Impact on users</td>
</tr>
</tbody>
</table>

Table 3.2: Criteria to evaluate artifacts (from March and Smith (1995))

The research goals of this study, as indicated in section 3.2, will chiefly involve outputs at the method level. The activities involved will be build and evaluate. The metrics and criteria suggested by March and Smith (1995) therefore are those of; operationability, efficiency, generality and ease of use. Among these, the stated research goals indicate that operationability is the main criteria, as so far as the method is intended to achieve operational functionality and hopefully offer some improvement.

Unfortunately, March and Smith do not explicate their suggested metrics, offering neither definitions of them, nor units of measurement for them. A number of these metrics bare close comparison with qualitative and so-called non-functional requirements such as those often associated in the literature with ‘softer’ aspects of system design, such as the UI and HCI. Such criteria are often considered difficult to
measure meaningfully, if at all, a position which Gilb (2006), a vociferous proponent of numerically measurable assessment, vigorously contests saying “… if I were emperor … researchers would focus on measuring what works and what doesn’t, and what things cost - and would research on real projects” and “all projects should be driven by maximum 10 quantified critical quality and benefit requirements”. The onus thus falls to the researcher, to develop workable success indicators for these objectives.

It appears there are no commonly agreed mechanisms for quantifiably measuring many of these criteria. Individual researchers and groups may select and develop differing sets metrics so long as they are able to indicate the ‘progress’ of the research project outcomes. Iivari (2002) allows that it may be sufficient to demonstrate, by a process called Ideational Evaluation, the novelty of an artefact, and its applicability to, and success with, difficulties (theoretical or practical) identified under pre-existing artefacts and their use.

A principle thematic concept of the preceding discussion is that, for design science research, the goal of the justifying investigation is to determine the efficacy of an artefact, rather than to seek to uncover its internal mechanisms or rationale. These issues of ‘how’ and ‘why’ are more closely aligned to the natural science research process of theorising and justifying. Whilst these are central issues in natural science research, for design science research they may be addressed after the artefact has been built and evaluated.

Theorising about design science artefacts produces explanations about how the artefact functions and performs as it does. Justification tests these explanations, usually against some form of analysis conducted against empirical data. These theories can be somewhat awkward to frame in some measurable and analytical manner. Norman (1986) for example, suggests that a quality of artefact performance may be measured by the divergence between how a user believes it works and how it actually works, whereby a small divergence indicates ‘better’ performance. Despite Norman’s admirable generation of a measurable characteristic, it may not be of high significance to the user if the final output of the artefact meets their needs. The degree to which an analyst abstracts phrases such as ‘how it works’ are of great significance here. There could be great difference between ‘how it works’ for a user and ‘how the user works it’. If the researcher is exploring ease of use or simplicity then such questions of divergence are highly significant. If poorly scoped or constructed then such questions may simply indicate the degree to which a user can guess or comprehend an artefact’s internal operating principles, such as a motor car’s internal combustion or a computer’s floating-point calculator. While such issues may be of interest, they lie outside the evaluation of research effort value or of artefact progress.

As the research goals of this thesis revolve about the construction of an innovative method and the evaluation of its performance against pre-existing methods, the March and Smith research framework is well suited. This framework has not been without its critics. Järvinen (1999) claims the framework is limited to the implementation of artefacts and that insufficient attention is paid to the development and preparation involved in building and evaluating. This criticism may be entirely
valid however Järvinen offers no evidence in support of it and it may reflect more a philosophical stance on the part of Järvinen than an established weakness.

The normative case-study framework does offer some guidance on the construction and evaluation of artefacts whose purpose is to improve some existing aspect of current methods. These are explored in the next section.

3.5 A Normative Case Study Framework

The previous section identified that the research goals lend themselves to a research framework in which a method will be both Build and Evaluated. March and Smith (1995) unfortunately offer neither a prescriptive process for building a method, nor for its evaluation.

A resolution is offered from the Research Goals, which clearly indicate an attempt to build the method upon a specific hypothesis, designed to improve some aspects of current methods. Such a notion suits the normative approach (Routio 2005). These approaches are distinct from more natural science modes of investigation, often called explanatory, exploratory, investigative or descriptive; as these seek to define the nature of a thing, what it is. Design science, as described earlier, of itself is not outside such descriptive research if the object is to define the designed artefact; unless the artefact is designed specifically to improve some existing condition. As improvement is central to the stated goals, then a descriptive research approach is less suited than a normative.

Investigating the research of artefacts, Routio (2005) offers that one outcome of a normative study is a detailed proposal of how an object be improved. Routio makes it plain that the object of such a study need not be a physical product, but may be a method or procedure.

In setting out a lifecycle for normative studies of this type, Routio (2005) offers the following four point process:

1. evaluative description of the initial state, defining the need for improvement
2. analysis of relationships and possibilities for change
3. synthesis of a proposed improvement
4. evaluation of the proposal

The stated Research Goals of this study included the four steps of Identifying challenges, Hypothesising bases for improvement, Construction of a method and Evaluation of that method. The process outlined above is clearly well-suited to these Research Goals. A normative approach would seem to be an ideal informing rationale for this study. Further, Routio’s (2005) detailed process is well suited to providing the operational details lacking from March and Smith’s (1995) more abstract research framework, and also from Vrazalic (2004) and Mwanza’s (2002) research methodologies.

Each of the four steps in Routio’s (2005) process are addressed in more detail in individual sections presented below.
3.5.1 Evaluative Description

The first of the stated Research Goals is “to identify challenges in existing SAD methods”. This is a close match to the first of Rutio’s (2005) four steps, which is to produce an “evaluative description of the initial state, defining the need for improvement”.

There are two approaches offered by Routio (2005). The first is an investigation of the ‘present state of things’ and the other a description of an ‘ideal state of things’. The former requires an objective description of the current state with some subjective appraisal of existing issues and challenges. The latter is constructed primarily from the subjective preferences of domain stakeholders and experts, typically in the absence of any usable model. Chapter 2 of this thesis produced a result more like the first of these approaches, especially as it was informed somewhat by the overview models of the domain provided by Finkelstein (2000) and Boehm (2006).

Routio observes that more traditional exploratory or descriptive studies are rarely deployed under a normative framework, except to establish a model of the exiting artefacts or conditions, against which some proposed improvement may be designed and evaluated. In such normative cases, the study is conducted under some evaluative criteria such as ‘usefulness’ or ‘satisfaction’. A final aim of the study is to “provide grounds for the planning of improvements to existing circumstances or products” (Routio 2005).

As normative research aims at improvement of the current state, on the scale of some evaluative criteria, the issues are necessarily subjective in nature. For this reason, Routio (2005) demands identification and consideration of whose opinions inform the model of the current state and the evaluation of any improvable aspects.

For the current study, the ‘present state of things’ is determined from a review the academic literature. Thus, in answer to Routio’s (2005) demands, the opinions are those of domain experts and established researchers. In so far as literature reviews are standard research tools under virtually all forms of research methodology, this means of investigation is justifiable. The suitability and breadth of those whose opinions are incorporated into forming a view of the ‘present state of things’ should also be agreeable.

This first stage of the research methodology, and with it the accomplishment of the first stated research goal, has therefore been satisfied by the contents and conclusions of the literature review (chapter 2) of this thesis. To briefly revisit these, a number of challenges and possible approaches to improvement were observed as follows:

- SAD and HCI methods alike tend to lack single, coherent informing philosophical or theoretical bases.
- SAD methods lack clear lines of communication between their phases (however they are arranged, or how tightly or loosely they are defined), to its side-bands such as HCI, and between analyst-designers and stakeholder-users.
Requirement capture is poorly understood and poorly conducted. There are grounds to consider selecting an informing theory which readily grasps the complexities and scaling issues of user organisations.

SAD methods tend to lose original focus or coherence and be adapted into expanded and/or *ad hoc* collections of practice because of high bureaucratic overheads, learnability and/or ease of use issues often resulting from conceptual complexity.

Current and *ad hoc* methods in practice rely too heavily on accumulated tacit knowledge and experience, making them difficult to transfer to new entrants and thus discouraging neophyte analyst-designers.

Some approaches to these challenges may be visible in the literature.

The DUEM method (Vrazalic 2004) for evaluating UIs suffered from too high an analytical overhead at the end of the project lifecycle, and too high a reliance on analyst expertise. An approach responding to this could have been, to shift the analysis of AT criteria to the design phase at the front of the project lifecycle, and incorporate the theory into simple design steps.

The design and specification of UIs and of the software systems they were for, appear as separate events, which suggested some possible utility may result from assembling a new analysis and design approach which addresses both what a system is required to do, in terms of how a user would operate it, hopefully capable of offering at least some specification as to interface.

The seeming lack of consensus regarding approaches to, and applications of, software development methods suggested motivation towards assembling the framework for a methodological approach informed by a consistent and coherent theory, which could prove useful, at least for some classes of software system.

A suitable theory was required to underpin this new method, and the seeming taxonomic trend towards activity orientation suggested an approach. A sizable body of literature suggests AT could have value in UI design. Application of AT had also been attempted in a systems design context, predominantly for highly interactive web-applications. AT might therefore, have been the best fit as the informing theory.

To summarise these: the proposal for improvement, which may be termed an hypothesis in the normative sense, involves the construction of an Activity Theoretic SAD method, addressing both system and UI specifications.

The next section details the means by which the second stage of the normative research process is conducted.

### 3.5.2 Analysis of Possible Changes

The second of the stated Research Goals is “to hypothesise some means of improvement to address the challenges”. This is a close match to the second of Routio’s (2005) four steps, which is the “analysis of relationships and possibilities
for change”. Routio observes that under normative research, an hypothesis is rarely used in the same manner as for informative or explanatory studies. Rather than being an unconfirmed candidate fact, a normative hypothesis is generally some principle or criterion which might inform the synthesis of an improved artefact or object.

Iivari (2003) expresses a similar idea when discussing design theory and design process. For Iivari, a design theory consists of some description of procedures for artefact construction, some informing kernel theory and a design hypothesis. This latter is used to ensure that the constructed artefact is consistent with the informing principles.

For Routio (2005), the most important aspect of an hypothesis is not to test it for some kind of truth as with an explanatory study, but rather to test it for practical value and functional operativeness. This is consistent with March and Smith’s (2005) research criteria of operationability.

As discussed in the literature review (chapter 2), there are considerable grounds to suppose that if a SAD method were informed and underpinned by a single consistent theoretical base, there should be an improvement in end-to-end consistency and verifiability of the end product against initially elicited requirements. It was also observed that a SAD method could be improved by incorporating within it some UI specification method.

Further, the literature review (chapter 2), proposed a trend towards Activity Orientation and attempts to construct specification and assessment methods for both systems and their UI, under AT.

The second stage of the research methodology, and with it the second stated research goal, may therefore be satisfied from the literature review (chapter 2) findings, by hypothesising that:

**SAD may be improved if conducted through a method constructed according to AT principles, which produces both system and UI specifications.**

The next section details how third stage of the research methodology, synthesis of a proposed change, whereby a SAD method is constructed in accordance with the hypothesis presented above.

### 3.5.3 Synthesis of a Proposed Change

It has been suggested that Software Engineering, as an activity directed towards resolving existing problems in software development, should seek to offer normative theories about software development (Tsai 1989). Simon *et al.* (1986) states that whilst a descriptive theory offers a statement about how a phenomenon ‘is’, a normative theory is one about how some phenomenon ‘ought to be’. Tsai [1989] defines a theory of the form “One should develop software using a new technique X” as a purely normative computational theory. Such an approach deploys normative principles in the construction of a new proposal, which addresses the build phase of the more abstract March and Smith (1995) framework.

The issue is then one of how to normatively construct an artefact. Routio (2005) offers some insight by describing two broad classes of design theory. The first is product specific, which seeks to produce a final artefact which conforms to some
well-established form. The second is goal specific which seeks to apply specific principles during the build, rather than concern itself with the final form per se. The latter is more closely associated with the normative process, as it can be used to improve rather than reproduce. If a product specific build process is correctly applied, then the artefact conforms to the desired form, which is to say it is what it should be, and under a normative framework, some test would be required to assess improvement. Under a correctly applied goal specific build process however, the qualities should be those of the goals set out by the informing hypothesis. It only remains to test if the artefact works (that is, if it can do what the preceding models could) to determine that the goals from the hypothesis can be applied.

In the current study, as has been established above, the hypothesis entails that an improved SAD method could be built upon AT. The build goal therefore is to adhere to AT. The test will be a test of functionality, which establishes that a SAD method could indeed be built on AT principles. As Iivari (2002) suggests, under his discussion of ideational evaluation, a criterion is for the functional artefact to be demonstrably novel. Several elements of the proposed activity theoretic SAD method would indeed entirely novel, assuming they could function as a SAD method. The notion that both system and UI specifications can be generated under a single theoretically consistent method is of itself novel. It will be necessary therefore to establish that a method artefact, constructed according to such novel principles and seeking such novel aims, is indeed a functioning artefact. Iivari (2002) observes that an empirical evaluation of the artefact’s qualities may be postponed for further, more descriptive studies.

In order to construct an artefact according to an informing principle, Routio’s (2005) notion of Iteration seems appropriate. Here the proposed artefact is modified and checked, then modified again. Routio however notes two weakness of the approach: Firstly, that iteration is generally only effective in handling one distinct feature of the artefact at a time. In the current study, whilst the hypothesis proposes that the artefact could offer multiple improvements, the common and distinct feature is that it be built and operated in ‘accordance to AT principles’. The iterative construction of the method artefact must therefore enforce adherence to AT principles.

Secondly, Routio (2005) observes that whilst iteration usually leads to a better solution, it is not guaranteed to find the optimal solution. The current study and it’s stated research goals makes no claim to optimality, nor does the normative framework require it. Future work will explore such issues, as foreshadowed by Iivari (2002). Neither weakness of the iterative build approach therefore impacts heavily on the current study.

In accordance with the preceding discussion, to satisfy the third normative step and the third stated research goal, the new SAD method will be constructed iteratively, in accordance with the principles of AT. It will then be necessary to establish that the method actually works. The next section details how the new method artefact will be evaluated.
3.5.4 Evaluating the Proposal

The fourth of the stated research goals is “To evaluate the new method against the identified challenges, by applying it against a realistic case study”. This is a close match to the last of Routio’s (2005) four steps, which is the “evaluation of the proposal”.

As discussed above, the proposed hypothesis entails that an improved SAD method could be built using AT principles. The preceding section detailed how that proposal may be instantiated as a method artefact, through iterative construction. The question which remains, therefore, is to determine that the resulting method artefact actually is an SAD method; which is to say, that it works. Furthermore, it must be able to demonstrate functionality across each of the areas identified in the challenges.

To establish that the method works, a test is required. Both Vrazalic (2004) and Mwanza (2002), who each proposed different AT inspired methods to improve upon some specific aspect of software projects as discussed in chapter 2, produced methodological artefacts and evaluated them by way of assessing claims they had made for their methods, under case study conditions. The stated Research Goals indicate that once constructed, the new method needs to be applied to a realistic case study in order to evaluate its performance against identified aspects of existing methods.

Thacher (2006) states that a normative case study cannot contribute to any explanatory or descriptive theory of how an object is, but rather offers contributions to some normative theory of how an object should be, evaluated under terms such as those employed by Aristotle, “the things that are good and bad for man”. Here Thacher (2006) makes it plain that the normative case study aims to contribute to our understanding of important values. Put simply in the terms of the current study, a normative case study can demonstrate that a functional activity SAD method may offer improvements against the identified challenges.

Under the case study evaluation, it will be necessary to establish that the novel method can do what an SAD method does. The novel method should therefore produce essentially the same result, being in this case according to Pressman (2001), sufficient information regarding the inputs, outputs and transformations of data within a system to allow for that system to be built.

3.6 Conclusion

As indicated in preceding sections, this research study is not a natural science investigation, and does not seek to explain or describe any phenomenon or object in nature. This study instead seeks to investigate and improve upon artefacts, inclusive of methods and processes, and as such is a design science investigation.

Under the March and Smith (1995) framework, this study will produce output of the form of ‘method’, and will conduct build and evaluate activities by way of its research effort. This framework, however, does not provide details of how these two activities may be conducted. It is necessary therefore to add a deeper layer of methodology.
As this study seeks to improve an aspect of existing SAD methods, it falls under the classification of a normative research study. Under this heading, the study will construct a method artefact and evaluate it for possible improvements over the existing methods.

Routio (2005) provides a four step process which is closely aligned with the stated Research Goals. These will be implemented under a normative research agenda. This results in the following four key stages of the current study:

1. Identification of challenges in both systems and UI design. This has been achieved by the literature review in chapter 2.

2. Proposal of a normative hypothesis for improvement against these challenges. The proposal to construct an activity theoretic SAD method for both systems and their UIs also emerged from the literature review in chapter 2.

3. The new activity theoretic SAD method is to be built, under a normative iterative approach.

4. The new activity theoretic SAD method is to be evaluated, under a normative case study approach.

The research methodology for this research study has now been established, as the four steps of the normative process, under the abstract definitions of build and evaluate. As indicated above, the first two steps of the normative process have been fulfilled by the literature review in chapter 2. The remaining steps will be conducted in the following chapters.
4.1 Introduction

In the previous chapter, a normative research framework was selected under which to achieve the four stated research goals. Those goals were:

1. To identify challenges in existing Systems Design and Analysis (SAD) methods.
2. To hypothesise some means of improvement to address the challenges.
3. To construct a SAD method, based on the hypothesis.
4. To evaluate the new method against the identified challenges, by applying it against a realistic case study.

Routio’s (2005) four stage process, as outlined in chapter 3, was chosen to address these goals. These stages are:

1. evaluative description of the initial state, defining the need for improvement
2. analysis of relationships and possibilities for change
3. synthesis of a proposed improvement
4. evaluation of the proposal

As explained in the previous chapter, the first two stages have been satisfied by the literature review (chapter 2). This chapter therefore addresses the third stage, the synthesis of a proposed improvement or under the March and Smith (1995) terminology, the Build phase.

After analysing the goals and challenges identified in the Literature Review (chapter 2), a number of tasks were outlined, central to which is the testing of the hypothesis that:

*SAD may be improved if conducted through a method constructed according to Activity Theory (AT) principles, which produces both System and User Interface (UI) specifications.*

The key objectives of the Build phase are to ensure that the method artefact built will:

- Consistently and coherently adhere to AT principles;
- Commence an AT informed analysis from the earliest phases of requirement elicitation and maintain adherence across the entire high-level lifecycle of the software project;
• Incorporate the principles of Activity Theory via simple prescriptive design steps in order to minimise the need for AT expertise on the part of the analyst/designer who will use the method.

An iterative approach was used to ensure that these objectives were followed. This involved assessment of both the underlying principles of the method as it was constructed and also of the nature of its output. Accordingly, the method was built in use, running against a suitable and well understood initial trial case.

The trial case, however, could not be addressed prior to there being at least some rudimentary components or conceptions of the method. Thus, it was initially necessary to establish broad concepts in the manner of March and Smith’s (1995) Constructs and Models, as described in the preceding chapter. The following sections present an historical overview of these earliest phases of method building. Reflecting the process of method building as it occurred, some of the following sections concern themselves with theoretical, or conceptual issues, whilst others concern themselves with experimental, or practical issues. Each new concept required some degree of practical trialling and each trial usually resulted in the need to re-examine, refine or further develop concepts.

### 4.2 Early Motivating Conceptions

As outlined in chapter 2, a persistent if not recently verified, perception exists that too many software projects fail. An equally persistent opinion in the literature suggests this may be attributable, at least in part, to failures in understanding, or meeting, users’ requirements. Some literature ((van Harmelen 2001), (Standish 1995), (Standish 2001), (Platt 2007), (Dix et al. 2004)) suggested a lack of concern for User Requirements have contributed significantly towards the commercial failure of many Systems. If a System is not attractive to users, and is thus not adopted, then it may be said to have failed, irrespective of its technical qualities. Likewise, if a System does not seem to perform which tasks users understand, expect or believe are necessary they may resist its adoption.

One area of SAD where users are of particular interest is that of the user interface (Dix et al. 2004). Early research interests in the principles of Human Computer Interaction (HCI) and Usability Engineering (UE) naturally produced sympathy towards a user centric approach. It sparked an interest in mechanisms whereby user requirements could be elicited, analysed and usefully deployed in SAD.

One enduring difficulty with users is that the details of their requirements are generally not expressed in any consistent or reliable fashion. Users and stakeholders generate their own notations and terminologies, complicating the business of capturing such details (Sommerville 1998). This difficulty informs approaches to requirement elicitation that are sensitive to consistency and viewpoint. It is necessary therefore to use a systematic approach when capturing requirements (Brown and Ghose 2004). Whilst user requirements may often be non-deterministic in nature, there are risks associated with formalising them. A significant risk of failure exists in marginalizing the stakeholder’s softer objectives, despite their inherent messiness. The pursuit of coherence and consistency must not be permitted to obscure, discard
or disregard those factors which, though often difficult to define or implement, may prove crucial to the success of the system under design.

One approach to identifying stakeholder preferences adopts a goal-orientation and asks ‘what does the stakeholder want to achieve’. Goal formulations express intended system properties (Lamsweerde 2001). The analyst will need to concentrate on issues of usability and identify these from stakeholder utterances, particularly for highly interactive systems.

Eliciting the requirements and setting out specifications, especially for the UI, of a multi-user system may be approached in a number of ways. There is no known deterministic process which yields globally ideal solutions. The problem, based as it is upon the vagaries of human behaviour, remains very much a subjective practice and something of a black art.

Experienced practitioners have often developed unique personal methods and techniques but seem unwilling or unable to share them. It seems in fact that these experience based methods are fundamentally tacit in nature and are thus extremely difficult to describe or transmit to the neophyte practitioner. Making the transition from beginner to experienced practitioner is therefore a rather hit and miss affair, which does little to assist the clientele.

Attempts to codify or prescribe methods for requirements specification have to date not shed much light on the process. A number of methods (as outlined in the literature review, chapter 2) provide descriptive and or quasi graphical tools for representing user statements and or for setting out specification schema, but little has been published which explains how the analyst transforms the one into the other. The existing methods generally appeal to experience or intuition for this crucial step. It is also observed that current methods tend to not offer end-to-end processes, to deploy consistent terminology, or to use any a solid and coherent theoretical base.

It has been commented indeed, by recognised experts in the field (personal conversations, Doctoral Consortium, OZCHI 2005, Canberra), that attempts to codify the experiential, intuitive and tacit art of requirement specification should perhaps be actively resisted. Their concern seems to be that prescriptive methods somehow stifle the intuitive and creative muse which informs and animates professional design practice. The method built in this chapter does not propose a solution to this quandary. Rather, it proposes a framework within which the analyst may use their own best judgement whilst still functioning within the informing theoretic principles.

There is no deterministic global solution to the specifying of a System to meet all requirements. It is unlikely that a number of analysts will produce identical candidate solutions or even conceive of the problem space in quite the same way. It should be recognised that a considerable variety of visualisations may be offered for any one System’s context and requirements. These will be incommensurate, in that it will not be valid to make a qualitative ranking of any one solution from the standpoint of another, since each is informed by a subjectively different conception of the problem space and therefore necessarily uses differing metrics for evaluation.
The motivation is not to devalue the hard-won skills of the experienced practitioner, nor to stifle their intuition and instinct. Rather, the motivation is to explore and understand at least one possible path from requirements to specifications in order to provide a viable starting point for less experienced practitioners. If a consistent, reliable, theory based and end-to-end method which will yield good candidate solutions to such non-deterministic problems exists, then at least the neophyte analyst can function in practice until their innate talents mature through experience.

4.3 Towards an Understanding of Design

The Oxford English Dictionary (OED 2004) conveyed a strong sense of *purpose* to the notion of design. A thing is designed to some predetermined end. The clear inference therefore was that the desired ends are known and may be expressed in the design. In SAD, these ends were most closely represented by the notion of Requirements.

Requirements might be divided into those things deemed Necessary and those deemed Optional, being those which cannot be omitted, and those which are desirable.

It is not always possible for all Requirements to be satisfied (Chung et al. 2000) as some may be mutually exclusive. Other requirements may be achievable to some degree, but might not be maximised except at the cost of some other. Such is the case with any problem with a form like that of a classic triangle inequality (Rudin 1976), where some solution may exist between the values of three crucial Requirements, none of which may be maximised without the negation of one or more others. Achieving an optimal solution under such conditions can become a Non-Deterministic problem depending upon the relative weightings of the requirements. Solutions to such problems can be of non-deterministic polynomial complexity, and may often become combinatorially explosive. Worst case design-type problems may even be classified as NP-Complete.

When the relative weightings of the requirements are non-persistent and variable across both time and the stakeholder groups individual contexts, then design might be said to become less a science than an art; as alluded to by those who champion the use of instinct, inspiration and experience.

This author offers a non-formal definition of design as follows:

>A successful Design is that set of Trade-Off decisions, which satisfy all necessary Requirements and an acceptably optimised set of Optional Requirements – within bounded constraints and limitations of time, technique and materiel.

So, the success of a system may depend upon the satisfaction of some set of users’ requirements, some of which may seem vague, inexpressible or even inconsequential. Further, there may be any number of potential solutions within the problem space, some of which are considerably more acceptable to the user than others for reasons that can be difficult to determine or calculate.
It could be useful if users’ requirements could be reliably elicited and factored into the design process. Perhaps these may be initially gathered from the expressed preferences of those users with direct concern for the proposed system. Taken as a set, such Stakeholder Utterances ($U$) could inform a design more likely to succeed.

If $U$ does not form part of the contiguous design process however, then possibility exists for divergence. Whilst it may be possible to elicit from $U$ some sort of direction for the desired product, intuition and empirical anecdotal evidence suggested that implementation could easily diverge during construction if progress were not somehow cross-checked against user requirements.

Figure 4.1 provides an indication of such divergence, where $U$ is gathered, and refined over time, whilst a set of Requirements Specifications is used to build a system. If the SAD process disconnects from $U$ then there is no mechanism for ensuring compliance with $U$. Divergence of this sort can prove sufficient for a System to result in failure, should the diverged product fail to meet with the user expectations expressed in $U$.

It is insufficient to allow a project to ‘run its course’ and then conduct validation and verification checks afterwards. Should the checks indicate failure the effort and resources have been wasted. If the checks are not based upon $U$ with sufficient fidelity, then crucial amounts of divergence may not even be discernable.

Figure 4.1: Divergence of implementation from elicited direction

It may prove insufficient for $U$ to be considered merely as a perfunctory ‘customer satisfaction’ exercise in the earliest phases of SAD. If analysis, design and implementation were insufficiently connected processes, the fidelity of the final product could be in question. Practitioners may have disregarded stakeholder utterances for a variety of reasons. Avison and Fitzgerald (2006) outline reasons why Participatory Design might be not be deployed such as their apparent non-expertise and their volatility. Analyst-Designers thus may have been unaware or unconcerned about the potential for divergence of their work from their clients’ expectations. It may, indeed, be insufficient for the analysis of $U$ and the development of the system to be disconnected if hopefully parallel processes. The elicitation and refinement of $U$ should perhaps be an incremental element of the analysis process, and the analysis
should be a seamless part of the design process. If analysis captures and accounts for crucial factors, and design is directly driven by these considerations then perhaps divergence can be minimised and verification and validation checks can be directly applied throughout the entire lifecycle, directly from initially elicited requirements.

It should be observed that some so-called extreme methods refer back to stakeholders after each incremental improvement to the system, and thus they could be said to have horizontal cross-checks against the users’ requirements. There is much to be said for this, however almost by definition it is neither a prescribed nor end-to-end workflow. It may be that some experience is required on the part of the analyst for best results. It may also be that by running elicitation concurrent with construction and these two strands influencing one another, the project could actually alter in scope and intent during the process, which can result in failure of strategic goals, cost over-runs and double handling.

In scoping this study, the issue of system usability and of the design of the UI became a focus because in this particular domain, the elicited requirements and needs of the stakeholders and users are accorded currency by a broader base of commentators.

Numerous methodologies exist for eliciting and recording requirements and generating specifications from these (see chapter 2); however it was felt that these were often insufficiently connected to the implementation and evaluation phases of the product lifecycle. Few, if any, methods applied a consistent theoretical basis, and a consistent suite of tools and techniques, across the whole product lifecycle.

4.4 Design Side Considerations

User centrism combined with the commitment to the principles of AT dictated under the research methodology from the outcomes of the literature review, suggested that a good starting point might be found in the literature at the intersection between UE and AT. As noted in chapter 2, perhaps the most appropriate, useful, closely related and useful example was the Distributed Usability Evaluation Method (DUEM) proposed by Vrazalic (2004). Usability Evaluation Methods (UEMs) however can only indicate quality ‘after the fact’. Sitting in a position analogous to Validation and Verification (V&V) (Zaphiris & Kurniawan 2006), UEMs make little impact upon design unless lessons learned from one project might be fruitfully applied to the next.

Vrazalik’s approach was to evaluate the User Interface (UI) of a system after it had been constructed following the precepts of both Distributed Usability (DU) and of AT. The method employed AT flavoured elicitation and analysis techniques not unlike those in Business Process Modelling (BPM), yet were deployed exclusively at the end phase of the lifecycle. Weaknesses of this approach included the difficulties of capturing a System’s requirements after the analysis and design and build phases are over, the complexity of applying AT to requirement elicitation (a process Vrazalic fails to make clear) and the low value in gathering evaluation data after a System lifecycle has completed.

Although the breadth of literature on evaluative methods indicates that post-build evaluation and V&V is valued by the community, it is likely that greater utility
would result if the SAD and evaluation were conceptually linked. If it is easier to elicit requirements at the outset and these could then inform more meaningful evaluations, then logic suggests that early-phase requirement elicitation should be of a form which can satisfy both processes.

It has been proposed (Brown 2004) that the AT analytic elements of the DUEM method may be more productively located at the earliest phases of the project lifecycle. If elicitation, analysis and design precede the central act of coding and building the software itself, and evaluation follows, then the decision was made to conduct the AT analysis on the ‘design-side’ of the process.

Gathering the AT styled requirements of a system and its UI at the outset might not only inform and possibly improve the design of both (according to the normative hypothesis) but also serve to inform an evaluation of the finished product, perhaps after the manner of DUEM.

Observing the trend towards partial automation of computer systems construction from Requirement Specifications (Boehm 2006), the possibility arose that someday the design phase and perhaps the evaluation phase may occupy a greater part of Software Engineering practice than the actual building of the product itself. Figure 4.2 illustrates one possible future in which SAD and Evaluation occupy a far greater part of the lifecycle than they seem to currently.

![Figure 4.2: A possible change of emphasis between stages of a product lifecycle](image)

If SAD and evaluation could become the most significant lifecycle components, and if their theoretical and conceptual bases could be strongly linked, then these processes together might comprise something of an architectural meta-process. Conceiving the lifecycle in such an overview fashion, in which construction is deemed a sub-task, promised potential.
Figure 4.3: An Early Workflow Concept

Figure 4.3 illustrates this early overview lifecycle conception as a crude workflow. This early workflow concept required the Specification of Requirements from $U$ via an, as yet undescribed, AT driven SAD method, but allowed for the final product to be tested against both the Specifications and the utterances $U$ under some consistent language. There was no understanding in this early conception of how the AT model might yield Requirement Specifications.

If AT could inform both SAD and evaluation, then it became necessary to frame and justify the proposal. Attention turned to the scope under which the proposal might have currency.

4.5 The Interaction Quotient (IQ)

As described in chapter 2, AT describes human activity. Its application to the analysis and design of Computer Systems relies therefore upon the uses humans make of them to enhance and facilitate their activities. As this is an exploratory and innovative study, it is reasonable to limit its scope initially to indicative cases where utility and improvement may be demonstrated. To build a method for designing Computer Systems of a kind which are closely associated with identifiable and describable Human Activities would suffice for this initial study.

Not all computerised systems exhibit the same degree of reliance upon exchanges with a user. Some systems operate almost entirely autonomously from human contact. The computerised fuel-injection system in a motorcar is just such a system – the driver may remain quite unaware of its operation. Some systems function in a batch-processing mode where they operate quite independently of any user, once initial conditions and commands have been set. Monthly salary processing in a large company may fall into such a category. Increasing numbers of systems, however, seemingly exist almost purely for the direct use of users and cease to function in any meaningful way without user interaction. A web-based e-commerce system would exemplify such a system.

To approach a metric for classifying systems with respect to these differences, we adopt the term Interaction Quotient (IQ) (Brown et al. 2005), which reflects the proportion of a system’s functionality which directly interacts with the user. It should be clear then that much, if not most, of the functionality ‘resides’ in relatively close proximity to the interface(s) of the system. We refer to systems as being high-IQ or low-IQ.
A greater number of high-IQ systems are being encountered over time as the World Wide Web becomes the medium for widespread access to systems; particularly those designed for use by a heterogeneous audience. A market for high-IQ systems exists in which we cannot assume any users’ prior computer skills. Ease of use is therefore a crucial factor.

The early motivational concerns regarding stakeholder/user acceptance outlined earlier (section 4.2) reinforce this focus on issues such as usability. As noted by (Maciaszek and Liong 2005), these aspects are commonly classified as Non-Functional Requirements (NFRs), being those which indicate qualitative factors. Under the common taxonomy, more concrete and operational factors are described by Functional Requirements (FRs). NFRs are perhaps most often seen in discussion of UI’s. (Chung et al. 2000)

As, by definition, high-IQ system functionality resides close to the UI, then specifying their Requirements necessarily pay close attention to these so-called NFR’s. A failure to perform against these NFR’s can often result in a failed system. The functionality of high-IQ Systems must therefore be closely associated with these so-called ‘non-functional’ NFR’s. The common FR-NFR taxonomy seemed insufficient and potentially misleading in these cases.

It was recognised that for high-IQ systems, a number of aspects traditionally considered to be NFRs are in fact FRs as, in this context, they described concrete and operational aspects of the System. A simple alternative taxonomy of was proposed (Brown et al. 2005) replacing the common FR-NFR dichotomy with the new terms, Internal Requirements and External Requirements. System elements, components or qualities which are visible to the user were labelled external requirements, and those which are not visible were labelled internal requirements.

These conceptual musings aided in scoping and justifying the study and seemed at the time to be internally and theoretically consistent. They also produced however, a skewed concentration of attention towards just the UI aspects of Systems. This misalignment of priorities would persist for some time.

4.6 Architectural Conception of the Software Lifecycle

As discussed in the Literature Review (chapter 2), there are several quite varied conceptions of the software lifecycle. Historically they have revolved about the construction phase and given considerably less attention to the SAD aspects. Often these are expressed in a few words as some kind of perfunctory process, a mere threshold to be quickly passed.

It is precisely the negative effects of not paying sufficient attention to these matters which inspired the current study. Accordingly, the motivating concerns of this study remain chiefly centred not on the actual construction of the system but on ensuring that the purpose which informs its design has been correctly captured, analysed and expressed. This study has no interest in how the final requirements are best implemented into executable software code. Such matters have been exhaustively researched in other venues for decades and, according to some sources (Boehm 2006) may even be making themselves technologically obsolete through automation.
Chapter 4 – Method Building

As the method under construction drew inspiration from the conceptually nearest AT-driven method, Vrazalic’s DUEM, it emerged that it could incorporate the notion of an end-phase evaluation. This yields the potential for a method which could operate from end-to-end, across the lifecycle.

It might seem odd to claim that a method which does not attempt to provide any kind of tools or techniques for implementation, micro-level detail or coding could be called end-to-end. It is the claim of this thesis that it is a valid claim, when the software lifecycle is viewed architecturally. The term architecture has been variously applied to SAD, usually to infer some form of modularity (often hierarchic) of software or System components and/or of their arrangement (Maciaszek and Liong 2005).

For computing in specific, the Oxford English Dictionary (OED 2004) offers as one definition of architecture: “The conceptual structure and overall logical organization of a computer or computer-based system from the point of view of its use or design” and in more general terms it refers to the derivation and use of an informing style or special method and to the abstract consideration of structure.

This study therefore adopts the following view: That a method may be considered to extend from end-to-end architecturally if the special method or abstract style is consistently applied to drive the design and confirm that the requirements have been satisfied. The actual details of implementation reside at a lower, subsumed, layer of the lifecycle and need not be addressed in detail.

4.7 How to Use AT?

Having decided that AT was a suitable candidate for building an architecturally end-to-end method, it became necessary to start addressing how that might be done. As discussed in the Literature review, no successful AT based design methods had been published, due largely to the inconstant nature of Leont’ev’s AT hierarchy of Operations, Actions and Activities. As some confusion was anticipated, some means for disambiguating the Leont’evian hierarchy was sought from the outset.

When investigating the numerous inter-related Activities of a stakeholder/user group, achieving a useful granularity of elicitation and analysis became crucially important. It had proven the stumbling block to some previous attempts to construct an AT design method (chapter 2). Previous investigations into Yu’s i* BPM syntax (Brown & Ghose 2004) suggested that it could be modified to assist with such hierarchic decompositions. Considerable effort into adapting i* proved fruitless when it was realised that work had diverged away from the principles of AT.

As noted in section 4.5 above, it was decided to commence with a consideration of the UI. At this early stage, work concentrated on the mechanism for how AT could extract UI Specifications from U. As Figure 4.4 shows, a nebulous set of stakeholder/user utterances (U) could be expressed as a network of Activities (represented by the Engström triangles) and then some as yet unspecified process, here tentatively named B*, would extract from the network sets of Actions which could somehow be conflated into a single coherent list of Requirements. It is interesting to note that even in this earliest conceptual stage, it was recognised that
more than one Activity would need to be considered, each being here represented by an Engström triangle. User actions would, therefore, need to include their interactions as well as there might be not only multiple users, but also multiple Activities per user.

![Figure 4.4: An Early Conception of an Activity Theoretic Analysis](image)

Having loosely defined the term IQ (Brown et al. 2005) as a measure of how interactive a system is, the study was scoped to high-IQ systems where the UI itself tends to form and reveal a large proportion of the systems functionality and purpose. In such cases, those external requirements (see section 4.5) often considered non-functional become crucial and the users actions might almost define the nature of the system if only they could be successfully elicited, analysed and applied. As it seemed it would be no more obvious complexity in having multiple users as multiple Activities, the scope could entail multi-user systems.

As a framework under which such actions might be analysed, AT promised much. It was necessary to re-examine AT and begin to see how it might be used to this end. According to the AT precepts described in the literature review, an Activity is the unit of analysis, being Motive driven, conducted by a Subject with the assistance of some Tools and producing some Outcome which might resemble the originally intended Object. Beneath that were Goal driven Actions and further still, atomic level Conditionally driven Operations.

As deployed in recent decades outside of Russia, AT applications (especially the work of Leon’tev (Kaptelinin 2005)) have resided largely in the realm of psychology and education (Johnson 2007), concentrating on exploring the Vygotskyian internal/external subjective nexus between desired object and the actual outcome. In the earlier literature reviewed for this study however, the tool node was typically recorded for context, to providing explanations of interesting object-outcome relationships.

Since the tool contains those physical, psychological and procedural facilitators which enable the subjects pursuit of his/her object, to adapt AT to a software engineering role, it became necessary to shift focus to the facilitating tool(s). The principle realisation was that the system itself is of course one of the facilitating enablers of the users actions. If AT could reveal the nature of the tool(s) used or needed, then this detail would necessarily contain the nature of the system. In a sense therefore, the method under construction would turn traditional AT on its side – focussing heavily on the tool node(s) and less so on the object-outcome nexus.
Even at this early stage it was understood that the tool node(s) would contain more than just the system, as AT itself recognises that numerous secondary, intangible and psychological processes and artefacts also reside there. Tool nodes therefore might contain such diverse details as the linguistic conventions and jargon used by users, their wall clocks, their work chairs, their scrap paper and their vocational skills. This as yet unspecified AT process would therefore need to capture and analyse the user’s actions, extract and conflate the tool descriptions and somehow extract and refine a system description from these. Whilst the notion of adapting AT for this purpose seemed straightforward if not appealing in principle, the reality was proving to be non-trivial.

It was hoped that an AT analysis of user interactions could yield a description of those tools (both material and psychological) which best facilitated the actions involved. From this description, a system could somehow be specified, particularly in terms of its Interface components, which would successfully enhance the multi-user activities. Although no concrete mechanisms had yet emerged, there was a notion that by collecting all the users’ Actions into one place might allow them to be analysed and sorted.

**4.8 Refining the Concept**

A more refined conception of the workflow for the method is shown in Figure 4.5. The need for some important new proto-mechanisms was identified.

Firstly, some means of deriving U from stakeholder utterances. Secondly, means for setting out the network of Engström Matrices. Some mechanism would be required to synthesize these utterances into a maximally consistent subset of activities, and indicate their network relationship.

At this early conceptual stage, the question of what it meant for Activities to be in some kind of a network was not addressed. What might comprise the arcs and nodes of such a network was not yet considered.

As with the earlier conception shown in Figure 4.4 a listing of Actions is extracted but here a new notion was conceived, whereby the list of Actions would be extracted from each Activity of the network and assembled into some form of tabular list. This was tentatively labelled the Combined Action Table (CAT), and remained a cornerstone idea despite having no structural or procedural flesh on its bones, and later undergoing some radical revisions.
The final stage shown in Figure 4.5 involves a diagrammatic representation of System components or, more specifically at that time, system UI components, each aligned to some particular activity identified from the network.

In this final stage, the diagrammatic representation of the actions was assembled, duplicated and mirrored to make a double list. These hierarchic action lists were to be extracted from the CAT, each now associated with its parent activity. This is an early expression of the notion that the hierarchy of system components may somehow correlate with the hierarchy of AT components.

It was conceived that somehow set of linkages between these actions could be derived from the CAT. They were visualised as dynamic transactions (being quite literally operational communications and relations between actions in the AT sense) where data, permissions, flags, obligations, notifications and the like were exchanged within the network of activities.

The intuition was that detailed descriptions of these transactions would somehow inform the UI at the point of each activity, and perhaps even give some clue as to the internal mechanisms of the system itself. At the terminus of each link should reside some kind of UI component which operates with (sends, receives, acknowledges etc) that particular transaction. Clusters of these termini could constitute, or at least indicate the nature of, the UI for that parent activity and the nature of the transactions themselves could indicate something of the system’s internal operations.

4.9 The Trial Case – Academic Assessment Task

The mechanisms suggested by the vague second workflow in Figure 4.5 offered some faint promise in so far as they hinted at a correlation between AT and system description, but lacked any form of rigour whatsoever and seemed to defy concrete description. With too few solid points on which to build, it was time to introduce a specific trial case. The trial case would provide a realistic data set and scenario against which to test-fit method fragments and concepts. In a very real sense it would serve as a kind of prototyping mannequin.

The trial case needed to be sufficiently realistic to indicate when the method components were working or failing, but needed to be sufficiently self contained to not overly confuse matters with too many variables. To facilitate easy access to genuine stakeholders and to allow the researcher-builder to retain a general familiarity with both the stakeholders and their operations, a routine academic process was chosen for the test case, namely: “Academic Administers an Assessment Task”.

The process stakeholders would be teaching Academics who each direct one or more Tutors. In this process, the academics design assessment tasks for students to complete. The tutor(s) may distribute, collect and mark some proportion of them. The academics collate and centrally register the results.

The method components would now be designed to fit this framework and, ultimately, comprise a method that could specify a system able to usefully facilitate the test case process.
To best explore the richness and potential of an AT analysis, the research study was scoped to monolithic (i.e., having one single broad agenda) multi-user systems, that are highly interactive. In so far as the test case suggested the need for a multi-user system which would have a highly interactive functionality (i.e., a high-IQ factor), it was a good match for the scope.

Preliminary peer review indicated that expert analysts and designers were not always receptive to any prescriptive design methods and could even go so far as to suggest that the ‘innate’ and ‘instinctive’ faculties of the analyst could be stifled by the use of any end-to-end method (pers conv, Doctoral Consortium OZCHI 2005, Canberra). Accordingly, the scope was more carefully refined to supply a clearly prescribed method which could allow neophyte designers to produce feasible systems irrespective of prior experience or tacit knowledge.

The UI-centrism of this early approach to building the method suggested benefits in further scoping the method to systems which could be designed and built by neophytes from common Graphical User Interface (GUI) elements. The trial case seemed a good fit to all these scoping constraints.

To investigate the method’s ability to cope with different interpretations of the process, two different Academics were interviewed, together with Students and Tutors of each. This produced, as expected, two quite different sets of utterances describing two different Activity sets. Basic AT principles suggested however that Subject-specific doings, with a common object should merge to a single activity, whilst those with identifiably different objects should form separate activities. Whilst it was easy to isolate the Academic as subject, identifying their objects and conflating their two distinctly different approaches proved a challenge. One recurring difficulty involved the variable granularity of AT.

4.10 Ambiguities in the Scoping of AT

As described in the literature review (chapter 2) Leont’ev introduced a sliding hierarchy of concepts into AT whereby a single human action or task might be deemed an activity, action or operation depending upon its context and or the degree of cognitive engagement involved.

This intriguing notion allowed Leont’ev to posit interesting observations regarding human cognitive involvement and the context of their actions and tasks but left an enigma for later researchers investigating the structure of the actions and tasks themselves. For example, might the task of obtaining a university degree best be represented as constituting one single activity or a cluster of closely related or even networked activities?

By AT definition, an activity comprises subject, object, tool(s), outcome and motive. Classic AT however does not provide as clear a set of components for the lower elements in its taxonomy, action and operation; nor for any higher abstractions, such as a Network of Activities. As might be observed from the text of the current and preceding sections, the potential for taxonomic confusion is non-trivial.

Verbal confusion aside, without clear conceptions of the diagnostic components of each hierarchic layer, nor any means of disambiguating the classification of any
human action into one or another, it becomes difficult to use AT in any deterministic manner. As reported in the literature review, Fjeld et al. (2002) baulked at this very hurdle and converted what seemed a potential system design method into a more classically AT flavoured anthropological analytical framework.

For the purposes of the current study, it would be necessary to overcome this difficulty. With hierarchic ambiguity embedded as a characteristic feature of AT, the issue required an approach sensitive to the flavour and features of classic AT.

Initially however one determination seemed obvious. If, according to AT precepts, the activity is the ‘unit of analysis’ then logically there should be at least one higher level of abstraction comprised of such units. Following the work of Kaptelinnin (1999), it was recognised that for complex multi-user tasks there may be a number of interrelated and interactive activities. It was decided that a fourth layer should be added to Leont’iev’s hierarchic model, namely an Activity Network layer.

### 4.10.1 Some Naming Conventions

As described in the previous section (section 4.10), attempts to explore the capabilities of this originally Russian-language theory could be confusing. The English language itself confounded discussion, where words such as activity, action, operation, task, deed, feat and so on are all close synonyms, increasing the potential for ambiguity and misconstruction. There were no clear terms to specify all the components of AT at all four of its layers.

It was decided to construct, arbitrarily, an extended hierarchic AT taxonomy, if only to facilitate progress in method building. Once words were chosen to fill these spots (see Table 4.1) it became necessary to have a meta-term for refer collectively to them. This was difficult as most common terms had been deployed in the taxonomy. Further, English lacked any collective noun for this set of concepts, as they were in effect the most abstract notion of verbs themselves. The somewhat irregular term ‘doings’ was adopted for this role, simply to avoid confusing misuse of any other existing English words. The Oxford English Dictionary (OED 2004) reports that the word had just such a use as far back at the 14th century, but that it was reportedly uncommon by the 18th century. To solve this taxonomic quandary, the study resurrected this antique usage of the word by way of a retro-neologism.

Arbitrary collective terms were required for the other aspects of AT, and thus the terms facilitator, driver, product and protagonist were adopted. Completion of the extended taxonomy, built to suit the Tool-centric focus of the study, revealed a number of Facilitators, or hierarchically abstract and refined conceptions of the Tool(s).

In drawing up the extended taxonomy shown in Table 4.1 it became necessary to explore the meaning of the highest level, that of a network of activities. If each activity has a subject and there are a number of activities, then potentially there can be numerous distinct and separate users acting as subjects.
Chapter 4 – Method Building

### Table 4.1: An Extended AT Taxonomy

<table>
<thead>
<tr>
<th>Doing</th>
<th>Facilitator</th>
<th>Driver</th>
<th>Product</th>
<th>Protagonist</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Activity Network</td>
<td>System</td>
<td>Agenda</td>
<td>Process</td>
<td>Group</td>
</tr>
<tr>
<td>3 Activity</td>
<td>Tool (Set of Screens)</td>
<td>Motive</td>
<td>Outcome</td>
<td>Subject</td>
</tr>
<tr>
<td>2 Action</td>
<td>Screen</td>
<td>Goal</td>
<td>Transaction</td>
<td>Role</td>
</tr>
<tr>
<td>1 Operation</td>
<td>Switch</td>
<td>Condition</td>
<td>Change</td>
<td>User</td>
</tr>
</tbody>
</table>

Whilst stakeholders, clients and users are accorded separate status under most SAD models, these distinctions may have less currency under an AT analysis as all may be deemed subjects or community members within a given activity. For the purposes of discussion these parties are here aggregated under the common term protagonist. Under the precepts of Engström’s socio-cultural “third generation” of AT:

- Protagonists with a common point of view on a given object comprise the Subject of a specific Activity,
- Associated protagonists sharing that object should appear in the community node of that activity,
- Relations between the subject and community members should be recorded in the rules and division-of-labour nodes.

Armed with this extended taxonomy, and the beginnings of some rudimentary heuristics derived from AT precepts, a return to the method concept promised to be more productive. Hopefully, the method could ultimately decompose stakeholder doings by analysing their drivers and products, and then identify the facilitators best suited to these. Careful analysis of the requirements gathered at each layer should permit recomposition of the facilitators at each layer until ultimately a system (the most abstract facilitator) was described. The description of the system would, for all practical purposes, form a feasible, defensible and consistent Requirements specification.

### 4.11 Components of the AT Model Based Method

With the terminology for a four-layered AT hierarchy (section 4.10.1) it was possible to return to the intuition that AT hierarchic components could correlate to system UI components (section 4.8).

As observed in section 4.10, there were unresolved issues of granular ambiguity inherent in AT. Whilst the concept had settled upon adopting a activity network abstraction at the highest layer (section 4.10.1), it was still dogged by the uncertainties of classifying activity level motives from action level goals or even operation level conditions.

Some degree of resolution was promised by the notion that the atomic doing component of the Users’ experience with the System was the discreet UI component.
The atomic AT component was an operation, conducted in response to some condition. The mapping between the two seemed obvious.

The taxonomy (Table 4.1) had arbitrarily termed the facilitator for a collection of operations a screen, which would appear at action level. A certain internal consistency with this mapping started to appear which increased confidence in this approach. Here a user might pursue a goal by conducting an action upon some UI Screen where all the necessary switches were gathered to conduct all necessary operations in response to various atomic conditions. The facilitator for the classic activity layer is, of course, a tool (or tools). There was potential value in the taxonomic intuition that this tool might be considered to be some Set of Screens.

**Figure 4.6: Mapping Activity Theoretic Elements to UI Components**

As Figure 4.6 indicates, the system itself (the facilitator at the highest abstraction activity network level) resides in the interstitial space between the activities conducted by each subject in the group. Each activity was therefore conceived of as residing at the ‘surface’ of the system, tangential to this System Space. The boundary surface between the system and each activity presented conceptual a location for the system’s UI for that activity.

As indicated in Figure 4.6, the UI surface onto the system for each activity was conceptually labelled a Station. The CAT now sat between a diagrammatic representation of the activity network located onto a system space, and a cluster of screens and switches. These latter could, it was hoped, constitute something towards a system UI specification.

### 4.11.1 The Five Ss

As illustrated in preceding sections, underlying conceptions of the method had begun to coalesce. The abstracted clustering of UI elements needed for any one activity, traditionally facilitated by a tool, was relabelled the screen set. The functionally atomic UI components which mapped to operations were labelled as switches. The cluster of operations, with their switches, which constituted an action would be facilitated by a screen.

The term station was used to indicate the location of any one given activity along the border of the interstitial System space. It was realised however that whilst an activity tends to have one Subject, any one individual user might be the subject of more than
one activity. The term station was thus refined to refer to the System Boundaries of a set of activities conducted by any individual user.

![Figure 4.7: The 5S Concept - the ‘Kidney Diagram’](image)

Figure 4.7 shows the relative arrangement of the System and its components. For the purposes of both mnemonic retention and market differentiation, all components were then named alliteratively. At this point, the Method acquired the name 5S, referring to its five principle conceptual components:

- **Switches** mapped to Conditional Operations,
- **Screens** to Goal driven Actions,
- **Screen Sets** to Motive driven Activities
- **Stations**, where Screen Sets are collected as needed according to the client organisational structure. It is a set of nested interfaces, needed for each of the Activities conducted by a given user. A user sits at a Station.
- The **System** itself is the computerised machinery behind and between all these UI components, residing in the space defined by the Activity Network.

Figure 4.7 illustrated the method concepts’ relationships, whereby one single station, comprising two activities and their screen sets, attached itself to the boundary of the system. The curious, if unintentional, overall graphic shape of this illustration, which served as a conceptual cornerstone from that stage onwards, earned it the engaging appellation Kidney Diagram.
Chapter 4 – Method Building

4.11.2 The Four Scratchboards
Combining the taxonomy from Table 4.1 and the 5S components from Figure 4.7, produced some vision of how the method might begin to work. Reconsidering the second workflow (Figure 4.5) in this light yielded the idea of four conceptual workspaces or ‘scratchboards’ wherein the system analyst-designer could do the method. Key analytical processes could be conducted in one or more of these scratchboard areas and produce the five part conceptual model, from which the requirements could hopefully be drawn.

There were:

1. **System Space**: the void between the activities of the network where the system would reside. This space could be described and initially bounded by defining the activity network. This would require identification of all the activities to be included. These activities would, by definition, be those whose tools included some elements that might be a part of the system. At that time no mechanism or heuristic had been proposed to do that.

2. **CAT**: the Combined Action Table (as it was at the time) could be produced by a decomposition analysis of the included activities. It yields connective flows of data, access and the like, between the actors and their actions.

3. **Patch Panel**: here a tabular representation of the hierarchic components of each activity could be assembled, duplicated and mirrored. It was conceived that somehow the CAT could be used to derive a set of linkages between actions. These linkages were visualised as dynamic transactions where data, permissions, flags, obligations etc were exchanged within the network of activities. The intuition was that detailed descriptions of these transactions would somehow inform the UI at the point of each activity, and perhaps even give some clue as to the internal mechanisms of the system. The patch panel’s chief function was to yield Switches, informed directly from operations identified from activities in the AT analysis of U.

4. **5S Model**: a kidney diagram styled representation of the entire system. This fourth main analytical scratchboard was actually a model of the overall product. It was an hierarchic representation rather than one of the processing steps per se. The 5S model was intended to accrete during the use of the preceding steps.

4.11.3 Grey Box Specification
A systematic AT-based Method for requirements elicitation and analysis should yield reliable and verifiable requirement specifications. The question arose as to what form, or degree, of specification might be expected from the method under construction.
Chapter 4 – Method Building

The notion of the method’s UI centrism became problematic as that suggested no more than a peripheral description of the system might be generated. As noted in section 4.6 however, the architecturally abstract approach would not delve into issues of micro-structure or coding.

Adopting a taxonomy of system-transparency often seen in literature about the testing of Service Oriented Architectures (SOA) (Buchi & Weck 1997), the UI-centric path threatened to yield only a Black Box level description which may not provide sufficient detail to facilitate System construction.

A micro-structure code-level description, never a goal of the study, would of course yield something like a White Box description where construction of the system is hindered and its true nature and key features obscured, by over specification.

It was hoped therefore that the method would ultimately provide a form of Grey Box level requirements, where only necessary details are exposed, comprising not only inputs and outputs but also descriptions of states, and indications of sequencing and performance.

Returning to the conception of the method, it was hoped that the linkages, recorded in the patch panel scratchboard, could not only indicate the atomic UI Switches, but also reveal something of the internal workings of the system. It would be necessary to not only indicate which actions joined to which other actions, but to describe the nature of the transactional flows between them in some way that captured notions of state, sequence and performance.

If this could be achieved the method would escape the bounds of the UI-centrism which had bounded its scope from the earliest conceptions. If feasible grey box specification of the system could be generated by the method, then it could lay claim to being a genuine SAD method. If this could be achieved, then under the terms of the normative research methodology described in chapter 3 the project would be a success. The method’s novelty and (hopefully) ease of use could make it an innovative and worthwhile entrant to the body of professional tools.

4.12 The Third Workflow Concept

Table 4.1, presented an hierarchic taxonomy for the conceptions used in the Method. When applied to the notion of the four scratchboards, this aided in describing a more detailed workflow process for the method.

A third workflow concept, shown in Figure 4.8, reflected the four hierarchic levels, here listed in ascending order, with item 4 being the most abstract:

1. Operation................................. Switch
2. Action........................................ Screen
3. Activity...................................... Tool (Set of Screens)
4. Activity Network......................... System

Some textual and iconographic conventions were adopted to encode as much information as possible into the figure. Square shapes indicated procedures, circular shapes indicated scratchboards, polygons indicated processes external to the method.
components under construction and the irregular ‘torn-page’ shapes represented output. Most of the elements in the figure were numerically annotated to indicate where they sat in the hierarchy.

Stakeholders were initially consulted during the ‘elicit #1’ procedure. The small number ‘4’ in the top corner of the procedure indicates that it starts at level four (activity network). This is because stakeholders can most readily their organisational context at higher abstractions. The large number ‘3’ indicates that questions asked in the ‘elicit #1’ procedure primarily targets information at the third level; that of Activities. The analyst extracts and refines level three data from level four input because the activity network is, of course, composed of some number of activities which need to be identified. The small number ‘2’ in the bottom corner of the ‘elicit #1’ procedure indicates that some level two issues will necessarily arise. The analyst should note down any and all preliminary level two results as they arise from this first procedure and use them later, when that level is more directly addressed. This notion reflects an important quality of AT, notably the awkwardly ‘slippery’ and amorphous structure of Leont’ev’s hierarchy as described earlier (section 4.10).

It became an important result in the 5S workflow from that point on that the layers were not, and could never be, discreetly separated from each other. Each layer would reflect upon information extracted in the more abstracted layer, and would give rise to notions in the next more refined layer. The analyst would be required, at all times, to retain a flexible approach wherein details at any level could be revised, adjusted, appended and refined at virtually any point in the SAD lifecycle.

The second and third elements show the scratchboard ‘moment’ where the level three (activity) abstraction is assembled and discussed with the stakeholders. This is actually the conclusion of the initial elicitation phase. This iterative nonlinearity of process would allow for mutual agreement upon the abstract conception of the organisational process presented by the 5S analysis. Iterative consultation of this sort is not unlike that found in business process conceptual modelling methods such as i* [Yu & Mylopoulos 1993 and 1994]. The scratchboard produced here loosely reflected the earlier ‘system space’ scratchboard notion, but in this third workflow was a sort of quasi-graphical map or diagram of the stakeholder organisational process and agenda, expressed in AT terms as a network graph of activities.

A second elicitation phase was envisaged, but the annotations indicated that whilst it was still building on the stakeholders’ natural familiarity with the highest (level four) abstraction, this one was specifically seeking to refine the second layer (action) detail. Similarly to the initial elicitation, the analyst is encouraged to note preliminary results for later use when considering level one.

The next element was a scratchboard ‘moment’, informed by the highest abstraction, but displaying mostly action data (level two). This was the CAT scratchboard identified earlier (section 4.11.2). By some, as yet undefined operation, this would allow construction of the next scratchboard, the patch panel where linkages between operational level doings would be exhaustively enumerated and classified. As described earlier, the hope was that these linkage descriptions could somehow be recomposed into the specifications for the system itself.
After production of the CAT, the third workflow concept jumps rather abruptly to the four key outputs; a list of switches (level one), descriptions of the screens on which they occur (level two), clusters of screens making up work-interfaces for each role (level three) and finally the requirement specification of the system itself.

Shown at the bottom of Figure 4.8, almost as an aside, are the concluding moments in the lifecycle whereby the System is constructed (a process entirely outside the architectural scope of this study, as described in section 4.6) and finally evaluated using Vrazalic’s DUEM Method described in the literature review. The reader may recall that one of the early motivations, described earlier, was to see if the AT principles used by Vrazalic could be adapted to the SAD phase such that a DUEM-like evaluation might occur with less overhead after the build phase.

The four scratchboards described in section 4.11.2 are closely matched by phases in the third workflow concept, but whilst they served as useful concepts, their role in a viable lifecycle.

Figure 4.9 presents an interesting compacted version of Figure 4.8, which was produced for a lecture presentation to a third year undergraduate HCI class. Observe that the scratchboard moment prior to the patch panel has now been clearly identified as the CAT. By this time, the workflow had been broken into four distinct phases; elicitation, decomposition, composition and finally, implementation and evaluation. A notable addition was the question of gathering the evaluation criteria from the three initial phases to be applied in the fourth phase, against the DUEM process. Importantly, the four output elements of the composition phase are here shown in a sequential processes, suggesting an ordinal and hierarchic relationship. The patch panel only directly informs the first three of these, with the final requirements specification arising from the preceding three outputs.
The third workflow concept indicated which layers of the AT hierarchy are being examined in each phase. It also introduced the first visualisation, however sketchy, of the recomposition (still referred to as composition) of the analysis into requirement specifications. Whilst some notion of what the analyst might be doing through the 5S method cycle began to emerge, there remained numerous open questions as to how most of these phases might be conducted. As these perfunctory details remained elusive, the next stage was to pursue some issues highlighted by the third workflow concept, namely; the mirrored hierarchy of both the AT model and of the recomposed outputs, the non-discreet nature of the layers, the relationship between decomposition and recomposition and the transformation of analytical data from one to the other.

4.13 The “U” Diagram

The third workflow concept illustrated in Figure 4.8 and Figure 4.9 and described in the previous section (4.12) suggested a parallel hierarchy between a series of decompositional or ‘analysis-like’ procedures and some series of recompositional or ‘design-like’ procedures, though these latter had been barely described.

Of the four phases evident in Figure 4.9, the decompositional and recompositional phases were the most important to examine. At some point there would have to be a changeover from an analytical set of deconstructive refinements, to a design-like set of cumulative abstractions.

The sequential nature of the third workflow suggested a natural turning point at the lowest level, where linkages between operations somehow informed the selection of
atomic GUI switches. In extended-AT terms, at this lowest level, users would bring about under certain conditions; a user could make a change by the operation of some switch. The obvious question was considered; if the patch panel at level one could be used to derive switches in level one, then could the CAT at level two similarly inform the specification of screens at level two. Perhaps the relationships might extend throughout the hierarchy, with the activity network (level three) directly informing the composition of screen sets. Hopefully the final requirements specification would show an obvious and clear mapping to the most abstract presentation of the stakeholders’ process and agenda.

The four layers of the AT hierarchy now seemed to lend structure to the core functional phases of the 5S method. The fourth workflow concept, shown in Figure 4.10, illustrated the notional relationships quite clearly. The workflow of the 5S method was now seen as passing twice through the hierarchic layers. The workflow passes through the hierarchy once downwards under decomposition and once upwards under recomposition. This curved path suggested the name U Diagram for this illustration.

It is important to understand that the boundary lines between the phases in Figure 4.10 are porous in both dimensions. As observed in the third workflow, there is some conceptual leakage between successive layers upwards and downwards. With the realisation that the path actually curved back upon itself, there are now conceptual leakages laterally from an analysis layer to its corresponding design layer. Aspects elicited at any given layer could now informed both the deconstruction as well as informing reconstruction as the workflow passed back through that same layer.
Chapter 4 – Method Building

The fourth workflow concept explored the point of turnaround between the two passes. There are only seven phases rather than eight because the 5S method converts conditional operations into atomic UI switches at the bottom of the U diagram. Once this crucial part of the process was recognised it became possible to assign tentative verbs to each phase in an attempt to approach the evasive question of how the 5S method would actually work.

However, a pitfall lay in this approach. The very phrase “map conditions to switches” would suggest a sideline in the method building process which would prove premature and ultimately a dead-end.

4.14 Getting Bogged Down In Details I

The fourth workflow concept illustrated by the ‘U diagram’ (Figure 4.10) centred on the mapping of operations to switches. AT theory dictated that operations are atomic doings, driven by conditions. If those conditions could be captured and described under decomposition, then they should be able to form the basis for the selection of suitable atomic UI components (switches).

As this mapping seemed to be the crucial process in the entire 5S method, a considerable amount of time was spent in an attempt to describe it. Initially this effort took an empirical form, drawing on the trial case ‘Academic Assessment Task’. Informal analysis of decomposed doings within the trial case was conducted on a micro-level. Attempts were made to find some deterministic heuristics whereby a given doing, such as ‘enter user name’ would yield a particular UI switch, such as an enter-text-dialog box.

Several micro-doings were examined in highly fluid and fast-paced whiteboard sessions without yielding any solid results. There seemed to be far too many unknown variables to approach any deterministic rules. After deciding that the size of this task was beyond the scope of the current study, it was decided to restrict the switches to an indicative subset of common GUI elements, sufficient for the normative trial.

After refining abstract concepts of the method over several iterations, it had seemed that one small deterministic module of the method had presented itself. It resisted easy resolution however. As a result, for a time the entire 5S method seemed best suited only for capturing elicitations in a stakeholder-friendly format. Whilst not a valueless result, it was far short of the stated goals.

In an attempt to work through the difficulty, much time was spent trying to formulate some commonalities. The scope of the work at that time seemed to shrink ever smaller. Each attempt to run a whiteboard analysis of the trial case seemed to require the ‘stakeholder’ to somehow already have some UI knowledge or know the solution to the problem.

4.14.1 Returning To AT

Having developed the 5S model concept as far as the ‘U diagram’ (Figure 4.10), the study seemed to strike an impasse striving to map conditions to switches. It soon became apparent that work had bogged down in this small area of detail.
A re-examination of the motivating goals and the driving philosophy of the study quickly indicated that attention had drifted away from AT. As it stood just after the realisation of the fourth workflow concept, the 5S method offered only the smallest nod in the direction of AT.

It was decided to set aside the question of quite how Conditions might map to Switches, although the notion that they should be mappable was not abandoned. The focus of attention moved to the first phase of the fourth workflow, elicitation of the Activity Network.

A return to the basic precepts of AT was attempted. First among these was the notion of the Activity as the base unit of analysis, and of Engström’s third generation activity matrix with its seven core elements; subject, tool, object, rules, community, division-of-labour and outcome, as described in the literature review (section 2.20.3). If AT could indeed form the basis of a useful SAD method, then its basic precepts should also present some utility.

It had always been a part of each conception of the method, that some series of formative questions would serve as the basis for structured initial elicitation from stakeholders. The DUEM elicitation questions posed by Vrazalic (2004) had been based upon the basic precepts of Engström’s third generation Activity matrix, together with Cultural Historical aspects. As such, they would serve as an excellent starting point and deserved to be examined.

4.14.2 The Elicitation Questions from DUEM
The fourth workflow concept of the 5S method seemed to offer considerable promise but had initially resulted in a developmental impasse. It was decided to return to the basic precepts of AT, specifically Engström’s third generation activity matrix (section 2.20.3) and the extended-AT taxonomy, and use them as the basis of first phase stakeholder elicitation.

It had always been envisaged that the first phase would deploy a structured series of questions designed to aid in identifying the most abstract layer (level 4) of the hierarchy, the activity network. The fourth workflow concept of the 5S method (section 4.13) dictated that such an examination should be expected to yield preliminary results in the next layer (level 3), that of individual activities. These questions are constructed from the elements and components of AT itself as identified in the extended taxonomy (Table 4.1).

Vrazalic (2004) followed a similar path in constructing elicitation questions for her DUEM method and it was considered that they could serve as an excellent starting point. Table 4.2 presents the DUEM elicitation questions from the Vrazalic thesis with commentary added during the current study. Each question was annotated with a commentary identifying its purpose, and where it might reside in the extended-AT taxonomy and which element of the Engström activity matrix it might be eliciting. Vrazalic made use of a broader set of AT related theory than the current study, and so some of her questions appealed to concepts outside the scope of the nascent 5S method.
As the comments indicate, of Vrazalic’s twelve questions, the first ten directly addressed third generation AT concepts. Of these, there seemed to be one for each of the seven main ‘nodes’ of the Engström matrix, plus two questions aimed at the action layer and one aimed at the most abstract activity network layer.

Question number one was problematic, as it seemed to be asking the stakeholder to be able to produce an AT-flavoured definition of from the outset. The problem may reside in some ambiguity with the question’s use of the term ‘activity’. Vrazalic could perhaps have benefited from defining a taxonomy to resolve such issues. This leaves question one hard to understand. Perhaps it is seeking for the highest level abstraction or perhaps it is eliciting individual activities.

At no point does Vrazalic attempt to identify the subject of the activity. Perhaps, for Vrazalic, only one single Engström-style activity is occurring and therefore the interviewee is the only candidate subject. According to Vrazalic (2004), these questions are asked of both ‘direct’ and ‘indirect’ users, which seemed to be Vrazalic’s terms for subjects and community members. This suggests that her ‘subject’ may be a group, but that there may still be just one activity. Her adoption of alternate terminologies is not clearly explained and leaves us with ambiguities.

It is difficult to see purpose behind the ordering of Vrazalic’s questions. Perhaps it would be uncharitable to suggest that their sequence seems arbitrary. Certainly Vrazalic’s thesis demonstrates a deep familiarity with AT and related theory. Perhaps some scoping ambiguity, so hard to avoid in the ‘sliding’ hierarchy of AT (section 4.10) has impacted the DUEM conception of scale in doings.

Vrazalic’s thesis demonstrated the DUEN evaluation method on the UI of a university website, to determine its efficacy in assisting some student (or students) to ‘obtain a degree’. In reading the thesis carefully, it is not easy to see if this rather complex, long-term and involved doing is treated as a single activity or a related cluster of them. Certainly the DUEN thesis does consider the complex doing “taking subjects / doing research” to be a single activity (Vrazalic 2004). The somewhat vague DUEN workflow, which is never set out in any stepwise or diagrammatic manner, seems to quickly depart from AT terminology. DUEN’s attention is focussed on issues of usability and more specifically on the related theoretical basis of Distributed Usability.

Whilst professing and demonstrating a strong adherence to the precepts of AT, it remains possible that Vrazalic was unable to make best use of the full power and potential of the theory. Although Vrazalic was aware of higher levels of ‘doing’ abstraction from the work of others such as Kaptelinin [1992], she seems to have envisaged and constructed her DUEN method to the earlier, classically Leont’evian three-level hierarchy. This cannot possibly be condemned as an error on Vrazalic’s part, although an opportunity to enhance the power, scope and utility of DUEN may have been missed as a result.

By not using a broader hierarchic scope, such as that offered in the 5S extended-taxonomy (section 4.10.1), any purposefully ordered sequence in Vrazalic’s elicitation questions is difficult to decode. They seem rather to somewhat randomly aim to capture all levels of the analysis in one pass. Further, there are some
taxonomic obscurities which suggest insufficient caution may have been applied to the perils of AT’s inherent scoping ambiguity.

It was decided that the 5S method would deliberately attempt to minimise scoping ambiguity issues. Initial elicitation questions would therefore specifically seek to identify the network of activities within the overall group process. This would, unavoidably, generate preliminary results for the next layer down. Questions would therefore embrace the opportunity to identify candidate activities which are, after all, the building blocks of the top layer activity network.

It was recognised that of all the elements addressed in AT, the only genuine invariants were the people, at least, those available during the analysis phase. The initial questions are therefore phrased to be largely person oriented. The identification of subjects serves as a readily available indication of activities, or at least, of some small set of related activities sharing some common protagonist.

Where the option exists, initial interviews would be conducted with those persons who have the most abstracted view of the process. If some agreed ranking hierarchy was available for the stakeholder organisation, such people were imagined as usually being nearer the top. This person might, or might not, be a part of the specific group that would ultimately use the system, but should certainly be aware of the group’s agenda, or of the organisation’s agenda for the group.

It is interesting to observe that at the highest abstraction (activity network), there may be evidence of an agenda imposed from without by the larger organisation, to which the group is related (as a subset or agency). Whilst this may be offered as an answer, as a stated agenda, it might not necessarily be the true agenda of the group. The true, or internal, agenda of the group might be considered as their chosen tactical solution, or response, to the problem, goal or puzzle set externally by the parent organisation’s agenda.

The initial level four elicitation questions produced for the 5S Method are presented in Table 4.3. It must be noted that the questions form a guideline for the analyst rather than a script per se. The Analyst would be encouraged to rephrase, repeat or even re-order the questions as necessary during interview. It was important that the initial elicitation be conducted in a manner sympathetic to the Group’s Agenda and values.

Despite the flexible applicability of the elicitation questions, the suggested sequence was quite deliberate. The structure of the questions strives to move from abstraction to refinement and from areas with more highly expected consensus, to more open and subjective areas. The objective was to maximise commonalities between elicitations from different interviewees. It was hoped that this could assist an analyst in achieving consensus more readily. By starting with more highly placed interviewees with more abstract notions of the process, this question ordering hopefully provides an agreed senior overview basis from which the analyst may customise further rounds of interview with persons in more specialised and targeted areas of the process.
<table>
<thead>
<tr>
<th>No</th>
<th>Vrazalic DUEM Elicitation Question</th>
<th>Commentary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>What is the key activity that a typical student at the university is involved in?</td>
<td>Asking stakeholder to describe the whole Engström Activity Matrix?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Perhaps seeking <strong>AGENDA Level 4</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Perhaps seeking <strong>OBJECT Level 3</strong></td>
</tr>
<tr>
<td>2</td>
<td>Why does a student engage in this activity? What motivates a student to be involved in this activity?</td>
<td>Seeking <strong>MOTIVE Level 3</strong></td>
</tr>
<tr>
<td>3</td>
<td>What are the goals of the activity and how are they achieved? How is the activity carried out? Are there different ways of performing the activity?</td>
<td>Seeking <strong>GOALS ACTION</strong>-level question <strong>Level 2</strong></td>
</tr>
<tr>
<td>4</td>
<td>What are the necessary conditions for the activity to take place?</td>
<td>Seeking <strong>CONDITIONS OPERATION</strong>-level question. <strong>Level 2</strong></td>
</tr>
<tr>
<td>5</td>
<td>What tools are used to carry out the activity? Are there alternative tools?</td>
<td><strong>TOOLS Level 3</strong></td>
</tr>
<tr>
<td>6</td>
<td>Who is involved in the activity?</td>
<td><strong>COMMUNITY Level 3</strong></td>
</tr>
<tr>
<td>7</td>
<td>What are the roles of the different people involved in the activity?</td>
<td><strong>DIVISION-of-LABOUR Level 3</strong></td>
</tr>
<tr>
<td>8</td>
<td>What are the rules that affect the way the activity is carried out?</td>
<td><strong>RULES Level 3</strong></td>
</tr>
<tr>
<td>9</td>
<td>What is the outcome of the activity? What is it used for?</td>
<td><strong>OUTCOME Level 3</strong></td>
</tr>
<tr>
<td>10</td>
<td>Are there any other activities which are directly related to this activity?</td>
<td>The entire <strong>ACTIVITY NETWORK Level 4</strong></td>
</tr>
<tr>
<td>11</td>
<td>Are there any problems in carrying out the activity?</td>
<td>Eliciting issues with the current process … Cultural Historical theory</td>
</tr>
<tr>
<td>12</td>
<td>Have you done this activity previously? Has it changed? If so, how?</td>
<td>Eliciting prior user experience … Cultural Historical theory</td>
</tr>
</tbody>
</table>

Table 4.2: Vrazalic’s DUEM Elicitation Questions - with commentary
1. What is the purpose of the Process? *(Agenda – level 4)*

2. Who is involved in this Group? *(Subjects – level 3)*

2a. What classes/roles of people are involved?

3. What does this Group do? *(Process – level 4)*

4. What do each of these people/classes/roles contribute? *(Object – level 3)*

5. What do each of these people/classes/roles produce? This may be expressed as their purpose or their result. *(Outcome – level 3)*

6. Why do each of these people/classes/roles carry out their Activity? *(Motive – level 3)*

Table 4.3: Proposed Phase One Elicitation Questions

At first glance it might seem counterintuitive that the analyst would seek to elicit the agenda of the process (question one) before establishing what the process actually is (question three). As described above, this ordering was based on the idea that people can often describe abstract motivating drivers more easily than provide a detailed breakdown of their doings. In addition, establishing more abstract notions may offer a quick contextual outline to assist the analyst in customising the rest of the elicitation process. The second question elicits details of involved people, as their identities should normally be available with minimal ambiguity.

The questions do not directly address all seven of the level three Engström activity matrix nodes as it was felt interviewees would not be able to meaningfully address all the nodes without familiarity with AT. The core priority with these questions was to facilitate a sketch of the activity network by identifying strong candidate activities. Those level three nodes most easily elicited were considered to be subject, object, motive and outcome. These, it was felt, would provide a sufficient basis for negotiating a consensual agreement on individual activities and some sense of their interplay.

As more highly placed interviewees with abstracted views of the process may not have accurate details of lower level doings and their drivers, some questions seek to identify further interviewees. It would prove necessary to ultimately conflate their statements a single acceptable activity network model of the process. The notion was to conduct elicitations in such a way as to support this synthesis.

Issues of rules and division-of-labour, it was hoped, would emerge during the negotiation to produce a consensual activity network map. The tool node, of course, is where the system component descriptions should accumulate. In this manner it
might be considered that the entire 5S method was striving to derive and conflate the contents of the tool nodes of each activity in the network.

To minimise the chance of missing crucial details, every potential subject or community member identified (or a representative member of the category or grouping they belong to) should also be interviewed if possible.

Each person identified and identified may be the subject of an activity, be a community member of an activity or perhaps not be associated with the process at all. This final category of people should ultimately not appear as subject or community member in any activity of the activity network.

Whilst the Phase One elicitation questions showed promise, and adhered closely to AT precepts, they left some open issues. Firstly, it would be necessary to cope with any variance between differing interviewees’ points’ of view. Secondly, it was necessary to express some heuristics to assist with identifying strong candidate activities for inclusion in the activity network conceptual map of the group’s process.

4.14.3 Heuristics for Identifying Activities, Subjects etc.

As discussed in the previous section, Table 4.3 presented a series of initial questions designed to elicit the activity network and identify candidate activities. It became necessary to declare some guidelines for identifying candidate activities and also for resolving with clashes or inconsistencies between different interviewees responses.

AT offers some insight into methods for identifying a candidate activity. Each activity will have a subject, a unique object, a motive and produce an outcome, which in turn can form an Engströmian ‘node’ in some other activity. Some of these indicative activity ‘node’ elements are considerably more difficult to confidently detect and declare than others.

AT’s inherent scoping ambiguity dictates that an activity is a flexible notion and depends upon how the analyst decomposes the objects. Whereas the DUEM thesis was prepared to take an exceptionally broad notion of activity, for the purposes of system design it is necessary to resolve down to the level of individual switches and so the analyst should seek to resolve far more finely cut activities.

Strictly speaking, activities are differentiated by their distinct objects, however, these are particularly difficult to identify and define. there remains an open question in the literature regarding the differences between object and outcome. Indeed, one of the intended functions of classical AT was to explore the object-outcome relationship between intentions, formed in the plane of internal reality and the resultant effect in the real world. Attempting to elicit these directly from stakeholders is fraught with ambiguity if even the analyst cannot be sure of their terms and parameters.

As with the agendas of groups at the highest level of abstraction, the motives of a subject can be difficult to capture. Here too, there may be both a stated or external motive and a more tactical or internal motive. This latter might never have been formalised or even put into words by the subject.
Subjects, however, are granular by definition. Being instantiated in individual human actors or in groupings of such individuals, candidate subjects are considerably easier to identify. However, whilst an activity has one subject (be that an individual or group), any one person, class, or organisational position or role may be involved in more than one activity.

An activity therefore was considered to be best defined by its ‘primary’ subject combined with its motive, so the individual (or individuals) for whom the motive has greatest currency would most likely be the true subject, and other mentioned individual(s) should be recorded as community members for that particular activity (though they be subjects of other activities). Indeed the analyst should expect, and look for, relations arising from just such interactions, as they tightly bind activities into the activity network and assist in later recomposition of switch elements into screens, screensets and stations within the system.

At first glance, an individual might be subject to several activities, and there might be several individuals acting as subject for a given activity. Therefore there can be some confusion between subjects in just such a part-whole relationship. In the trial case ‘Academic Assessment Task’ it was found that the interviewed Academics referred to other individuals acting in the role of Tutor. This seemed to present a strong candidate as the Subject some one or more activities in the activity network. Some specific examples of Academic Assessment Tasks discussed during elicitation referred to a single Tutor, whilst others referred to the use of a number of Tutors.

It was considered necessary therefore to express some heuristics for dealing with situations of this type. When should ‘Tutor’ (singular) be deemed a separate type of subject to ‘Tutors’ (plural). The situation expressed the potential for confusion between subject groups and subsets thereof. The following simple heuristics were proposed.

- If the doing of the subset *can* be conducted in the absence of the rest of the subject group, then the actions within the activity must be designed to allow for some or all or the subject(s) to conduct the doings individually as required.

- If the doing of the subset *must* be conducted in the absence of the rest of the subject group, then the activity needs to be split into two or more activities; one in which the activity is conducted by the entire subject group, another conducted by some subset of the group.

- Or, it may be necessary to create an *entirely new Subject*, consisting of some part-whole subset of the previous subject group (and possibly others), essentially a new role, for this activity and possibly also for some other related activities.

Each activity in the network will have a set of level two actions, suggested elements of which may be elicited from the initial analysis. A prime example of the inherent scope ambiguity in AT is the potential for actions to be promoted, under consideration, to activity and for actions to be demoted to Action. It is important to note that interviewees at different hierarchic levels of the parent organisation may
have markedly different opinions on the ranking of specific doings. The analyst must keep a flexible approach and perhaps maintain more than one candidate map of the Activity Network. Ultimately the need to identify atomic operations on the system (that cluster of computer-moderated tools which facilitates the doings of the activity network) and the switches that these might map to should guide the analyst.

As has been seen, interviewing stakeholders may often give rise to multiple conceptions of the activity network. AT has considerable potential as the basis for facilitating discourse, on common terms, towards consensus. The deployment of a common suite of descriptive concepts and terms within a theoretically-defined taxonomy creates a framework within which differing expressions from individual stakeholders may be compared on a more nearly equivalent basis.

It was anticipated that, once expressed less ambiguously in near equivalent AT-flavoured terms, the previously vague set of stakeholder utterances (U) would emerge as a valuable resource and basis for ongoing analysis.

Recall that traditionally, a main purpose of AT is to identify object of an activity. Identification of objects is a crucial aim of AT when deployed to examine issues of human cognition, in such disciplines as education theory. This research however, was more interested in isolating each individual activity out of U and representing these as a network to aid in identifying and defining the characteristics and boundaries of a potential computer-driven mediating system.

Since stakeholders lacking a background in AT’s theoretical concepts rarely conceive of their doings as a set of inter-related Activities. The set U however would likely contain numerous utterances phrased in terms of roles and goals, where goals are those consciously expressed aims that are finite and satisfiable (unlike objects). According to AT, any goal identifies an action, but it remained unclear how best to identify activities. Indeed, the bulk of published AT case-studies examined in this research bemoaned just how difficult activity identification can be.

It was decided at this stage to pay greater attention to activity outcomes than to their objects. Outcomes were envisioned as a sort of concretised product of a set of goals.

At this point in the research, the rule, community and division-of-labour nodes of the Engström matrix presented limited potential. Collectively they were considered to simply explore the socio-cultural-historical context under which objects are formed. The analytical utility of examining and analysing Engström’s rule and division-of-labour nodes was considered minimal at this stage.

4.15 Looking At the Issues Again: The Relational Database

Some confusion arose as to precisely what useful data could be extracted from U and how it should best be analysed and processed. Expressing U in AT terms seemed to yield multiple versions of similar data without any clear pathways towards the identification of Activities and their inter-relations.

To address this difficulty from a fresh perspective, it was decided to review the various information and data that might be elicited or required. To assist in
organising these thoughts, an attempt was made to imagine what form a relational database might take which served to facilitate such an analytical method.

With a view towards the type of portable tools a systems analyst might prefer to use when visiting a stakeholder’s business, a low cost mid-ranged relational application from HanDBase was selected and deployed on a PalmOS PDA.

Figure 4.11: Basic Entity Relationship Structure

Using the HanDBase application on both a PalmPilot PDA and an MS-Windows™ based desktop PC, some steps were taken towards building forms and tables to capture the information an AT-flavoured elicitation process was expected to uncover. It should be noted that this was never intended to be a fully developed AT-inspired business process analytical support tool, but simply an exercise in identifying, locating and classifying the information generated by the method as it stood at the time.

Figure 4.11 shows a basic entity relationship styled representation of the proposed relational database. Consistent with the unclear utility of information stored in the Engström nodes, role and division-of-labour noted above in section 4.14.3, these data were simply assigned a collective placeholder function as generic ‘constraints’ upon the activities.
Figure 4.12 through to Figure 4.15 illustrate some sample screen captures of the database forms created in this process. Some sample data from the trial case was entered into these forms simply to test functionality.
4.15.1 Roles

As a direct result of attempting to arrange the Engström–node styled data into a relational database, and considering the situation explored in the trial case, it was observed that a many-to-many relationship was possible between activity subjects and the actual individual people who perform those doings. Any individual might act as subject in a number of activities, and some activities within the stakeholder organisation might be performable by multiple individuals.

The common database design solution to such situations is to insert a join table, which mediates the many-to-many relation as two many-to-one relations. Figure 4.16 shows a sample screen capture of the role table, inserted between the subject and people tables.

The notion of roles would later prove to be quite useful in investigating how best to cluster the system’s interface widgets according to activity and ‘user’. This seemed to be the best candidate to serve as an indicator for the interface ‘stations’, as described in the 5S conception (section 4.11.1) and illustrated in the Figure 4.7 kidney diagram. The role, and its station interface ‘cluster’ was now a container for the coincidence of individuals acting as subjects.
It was realised that the coincidence of individuals in roles might not be invariant across different stakeholder groups even if they perform the same overall process with the same agenda. Indeed, even within a given stakeholder group, this coincidence may change over time as individuals join and leave and the business process is restructured.

This realisation indicated that the method had the potential to adapt to a number of organisations with similar processes and agendas, as the more refined components could be quite similar, and the chief differences may only arise at the station level, allowing for effective re-use of specifications. It also allowed for the potential to be readily re-engineered when stakeholder organisations were restructured.

### 4.15.2 Rules and the Division-of-Labour

The database exercise had thrown the seeming redundancy of rule and division-of-labour (DivLAB) data into sharp relief. They had been simply collected into a constraint table, awaiting determination of their function, if any.

The sorts of information which the trial case could generate for this table included notions of ordinality and dependence. Stakeholders were likely to suggest that a given doing required the outcome of some other earlier doing, or that the resulting outcome of a doing may need to be distributed to several other doings.

Since a primary goal at this stage of the research was to uncover how activities could interact and inter-relate in a network, it became apparent that the rules and DivLAB tended to indicate the presence of other doings (hopefully at the activity level) and could record operational constraints imposed by the larger activity network structure and the dependencies between each activity.

For example: in the trial case “Academic Administers an Assessment Task” the Student’s first important doing was to “get assessment task questions”.

In considering this particular doing, a rule became apparent. Namely that:

- The assessment task questions would be available only after they had been written and the student had been notified of their availability; which inferred there must be some doing required of the Academic or Tutor to issue a notification

This suggested the existence of a doing which had, as yet, not been identified. The identification of a rule had contributed directly to the identification of a candidate activity and more significantly, suggested an inter-relational linkage.

It became clear that rules and DivLAB could indicate or infer the existence of other activities and describe their linkages. Exactly how to proceduralise or automate the elicitation of Activities from the collection of rules and divisions-of-labour had not become clear but the potential for such a mechanism was obvious.

An early notion was to borrow some terminology from Formal Methods and attempt to interpret or translate rules and DivLAB into formal clauses expressed with the logical operators of sequence (temporal logics) and of dependence and obligation
(deontic logics). Accordingly, the entity relation diagram was changed to reflect this concept though the database structure as yet reflected no significant change because the mechanisms for adapting formal methods to this method were not yet determined.

![Basic Entity Relationship Structure with Roles](image)

Figure 4.17: Basic Entity Relationship Structure with Roles

Figure 4.17 shows the entity relationship expanded to include some notion of the potential utility of rule and division-of-labour data, as well as the role table, inserted to mediate the many-to-many relationship between subjects and people.

The notion of expressing AT concepts and conducting AT-based analysis with tools taken from formal methods presented considerable allure. The potential for introducing an unambiguous flavour of discrete mathematical ‘rigour’ to the method should surely contribute to its positive reception among the traditional SAD, SE and requirements engineering (RE) communities.

There was, as yet, no suggestion of any mechanism for extracting such promising logical operators from the vague set of utterances, $U$. Since AT should allow for $U$ to have a consistent taxonomic basis, perhaps it could be readily adapted to facilitate formal notation as well.

### 4.16 Getting Bogged Down In Details II

Consideration of the utility of rule and DivLAB data had strongly suggested their great value in uncovering Activities and defining their interactions. The potential ‘market’ benefits of adapting some discrete mathematical elements of formal
methods presented a powerful lure. Together these redirected the research into something of an intellectual cul-de-sac.

Since AT-styled elicitation generally produced natural language statements from stakeholders, efforts were made to explore how best to extract logical operators from natural language utterances. This is, of itself, an immense, rich and ongoing field of research well beyond the scope of the initial research question.

The notion of delving so deeply into formal methods, which would fundamentally restructure and relocate the entire research program, was shelved and flagged as a potential ‘future work’ point. Attempts to identify more basic mechanisms however had meanwhile uncovered the notion of Scenario Based Design (Rosson 2002), a notion more commonly associated with HCI (Dix et al. 2005) and thus closer to the initial research motivation.

This reminder of the original interface-centric motivation prompted a brief reconsideration of the larger method structure. The then current fourth workflow model, illustrated by Figure 4.10, the U diagram, hinged crucially around the derivation of individual interface widgets (switches) from the most basic operation-scale doings, as extracted from the AT informed decomposition of the stakeholders’ doings.

As the core function of method under construction would ultimately be to decompose stakeholder doings and identify the facilitators best suited to these, it was hoped that a scenario flavoured mechanism could assist in translating operations directly into switches.

Whilst possibly rather naive, the hope that such a direct mechanism could be produced gave rise to the notion that careful analysis of the requirements gathered at each layer should permit recomposition of the various facilitators (switch, screen etc) at each layer until ultimately a system (the most abstract facilitator) was described.

This describes a notional variant of the U diagram (Figure 4.10) in which some sort of cross-checking mechanism could be found at each conceptual AT layer. It was hoped that in some future work, the formal logical operators suggested by rule and DivLAB data could directly inform, or at least confirm, the recomposed system elements at higher levels.

In this way, the potential existed for the method to contain evaluation, validation and possibly even verification mechanisms, directly addressing the gap identified in the lecture-version of the third workflow model (Figure 4.9).

**4.16.1 Story Fragment Analysis**

As described above, some time was allocated to the notion of finding how switches might be best chosen directly from the descriptions of the conditions (which are the drivers for operations). This effort took the form of attempting to construct a condition-switch matrix, from which key descriptive terms could directly indicate suitable interface widgets. This produced an augmented U diagram (Figure 4.18) in which the process of directly translating operations to switches played a crucial role.
In this conception, some body of transactions operated between stakeholder activities. Via some undefined transformational process these transactions could be rendered into AT terms, decomposed to the operational level, translated into switches, then recomposed, and transformed into activity level transactions.

Transactions, it was imagined, would be elicited in natural language and parsed for some standard lexical tags. In effect there would be a series of little narrative stories which described each relationship between activities. Such a verbal, interview-based method would sit well with traditional applications of AT.

The collection and analysis of stakeholder utterances using some form of standardised lexical tags suggested a new mechanism for evaluation in the final phases of the proposed method. Many existing methods make use of the concept of scenarios for design choice, design justification and occasionally for evaluation.

Since stakeholders produced essentially narrative answers under elicitation, and since it was hoped that direct translation between conditions and switches was possible, it was likewise hoped that some form of scenario-inspired narrative mechanism could serve to evaluate, validate and even verify systems produced by the method under construction. This short-lived notion became known as the Story Fragment Analysis (SFA) concept. It did not survive long enough to appear in either published or public seminar formats.

It was hoped that once the requirement specification was composed, it could be possible to assemble almost any normal user scenario from the fragments of transactional statements. The SFA notion allowed for the system to be checked by determining if any normal test scenario could be mapped into a sequence of lexical tags drawn from the requirements specification. In effect, if any normal stakeholder scenario could be ‘assembled’ from tags contained within the system specification, then that scenario should be catered for by the functionality of the system.

This intriguing notion seemingly allowed for both a validation of the design, by running SFA checks against the system specification, and also for a validation of the instantiated and implemented final system.

Any inability to trace out a given stakeholder scenario should indicate both that the specification is incorrect or incomplete and also that some re-examination is required. Of course, it could also suggest that he posited scenario is of an exceptional nature and beyond the stakeholder’s definition of ‘normal’ activity. This stakeholders’ definition of ‘normal’ activity should be contained within the conflation of the activities and actions they described under elicitation.

Under this rather optimistic ‘natural language based’ vision of the method, it was hoped that sufficient negotiation between the analyst and the stakeholders in the earliest phases of the decomposition should result in mutually agreed definitions for all ‘normal’ activities. If all mutually agreed activities were then decomposed into Actions and then further into operations, then the UI elements recomposed following the cues from the higher layer elicitations, then the specification should be acceptable. If the system were built in accordance with the specification correctly,
then it should meet the requirements of all possible variations of ‘normal’ group activities, and thus should have successfully facilitated the group’s agenda.

One candidate identified for investigation was Create, Read, Update and Delete (CRUD) (Kilov 1998) where the functionality of systems were described under these functional headers.

Using the CRUD concept as inspiration, time was spent attempting to collate all the potential screen elements and map them to conditions, however the effort necessary to form a such a comprehensive list was to prove to be well beyond the scope of the research.

Instead, an attempt was made to assemble just an indicative subset of Switches, sufficient for a normative proof-of-concept in both the exploratory trial case and the final test case. The scope was recast to limit the UI to the common Windows Icon
Chapter 4 – Method Building

Menu Pointing Device (WIMP) Graphical User Interface (GUI) paradigm, which is well understood, readily available, widespread and easily coded using common visual-code construction tools.

As the method begins by eliciting statements from stakeholders, which are structured to map into Engström’s AT nodes, it became apparent that such an open style of response could be difficult to record with precision. By mapping the utterances not only into AT, but also onto a standardised lexicon of transactional terms, the analyst’s task could be greatly simplified.

It was anticipated that if system specifications were recorded under a standard transactional lexicon, they should be open to re-use and later modification by another analyst, if familiar with the method and its terms. Further, if well structured, the lexicon itself should serve to facilitate elicitation and decomposition.

Transactional relations, once identified at the action level in the patch panel, if phrased under a standard lexicon, should have been readily mappable onto the indicative subset of WIMP/GUI Switches.

For example the transactional statement: “Chooses one form of address, from a list of less than five options” may have suggested the use of a small panel of radio buttons. The exclusivity inferred in the transaction statement would have ruled out the use of check box type elements, and the small number of fixed choices would have suggested an interface element where all options are visible at a glance, rather than compacted in a scroll-bar or pull-down menu. Likewise, the pre-configured options mentioned in the transactional statement should have precluded the need for any open entry element, such as a text-entry dialogue box.

Of course there may be more than one standard Switch element suitable for any given transaction, and the choice would reflect the personal leanings of the analyst (and/or the stakeholders, if asked), but at least the neophyte analyst could be confident the design decision would suit the task in hand. The design would be functional, appropriate and feasible at least, if not necessarily elegant.

4.16.2 The Standard Transactional Lexicon

To form the Condition-Switch Matrix, run the Method and conduct validation under the SFA conception, required the assembly of a standard transactional lexicon, sufficiently comprehensive to facilitate an indicative testing of the AT based Method.

Once equipped with a standard transactional lexicon, the analyst would actively listen for these (or sufficiently similar) terms and then seek to clarify & confirm them with the stakeholder(s). The lexicon was intended to actually guide the analyst’s elicitation by suggesting terms for prompting the stakeholder. The use of the lexicon during elicitation might be imagined as parallelling the use by Police of facial image fragments to assemble a recognisable likeness for identifying criminal suspects when eliciting statements from victims of crime. Ultimately each element at each of the four levels of AT analysis would have had a narrative-fragment description, a short sentence-fragment that described the entity using terms drawn from the lexicon.
To test the scope or completeness of a system designed under this narrative-scenario concept, the analyst might to ‘run’ any normal stakeholder activity through it. As a part of the design process, some narrative fragment would have informed each component. Any normal stakeholder task should be expressible as some collected set of these informing fragments; that is to say, the capabilities of the system should accommodate any normal stakeholder task. Should any normal stakeholder task prove inexpressible in this way, then the system design would lack capability, most probably arising from an insufficient initial elicitation of requirements. Activity oriented sentences and phrases could be assembled from the lexicon, and these would aid in elicitation, design and ultimately in assessment. This natural-language ‘mosaic’ was at the heart of the SFA concept.

It was observed that the SFA lexicon would probably need slightly different levels of granularity at each AT level. Operation level switches would include standardised WIMP/GUI widgets and the terms and arguments which indicated their selection; whilst at the action level, more abstracted terms could collectively described whole classes of condition-switch pairing. Such action level terms would have been Goal oriented and transactional in nature, whilst the operation level terms would have been change oriented and conditional in nature.

The first step towards building the SFA lexicon was the identification and collection of a broad category of condition cases (essentially verbs) at the Action level. For example the term “open” may include a category of conditional cases such as “import” “create” “new” and might itself also suggest a check against a “choose” category (ie: open what from where?) and suggest the deployment of some kind of “choosing” interface element, such as a list-box or pull-down selection widget.

It was imagined that the SFA lexicon would present a lattice-like syntactic structure where a relatively small number of high-level “command words”, expanded from those presented in the CRUD set, could be decomposed into lower layers, ultimately suggesting the adoption and deployment of a finite number of candidate interface component switches (widgets).

A brief initial attempt to expand the CRUD set and identify a comprehensive set of core command words produced the following:

```
MAKE, CREATE, GET, FIND, PICK, DOWNLOAD, IMPORT, INSERT, SELECT, MODIFY, EDIT, RECORD, DISPLAY, EXAMINE, CONSULT, WATCH-STATUS, CLASSIFY, STAMP, SIGN, VERIFY, VALIDATE, REPORT, COMMUNICATE, DISPLAY, COMPARE, SEND, PUT, STORE, SORT, SEARCH, SPECIFY, SECURE/UNSECURE.
```

A similarly brief attempt to identify classes of GUI Switches produced:

```
BUTTON, LABEL, TEXT BOX, MENU ITEM, CHECK BOX, RADIO BUTTON, PICTURE BOX, PANEL, DATA GRID, LIST BOX, COMBO BOX, LIST VIEW, TREE VIEW, TAB, CONTROL, HORIZONTAL/VERTICAL SCROLLS, TIMER, DOMAIN UP/DOWN, NUMERIC UP/DOWN, TRACK BAR, PROGRESS BAR,
```
An early attempt to simply the glossary by identifying suitable common meanings for command terms produced Table 4.4 (below). The problem was one of deliberately simplifying natural language to some commonly acceptable and near-universally applicable glossary. This rather ambitious goal would have involved far more effort than the project could justify, given that not dissimilar efforts inform entire schools and programs of academic activity in fields such as ontology and the semantic web (Berners-Lee 1998).

It was hoped that ‘sentences’ drawn from the SFA lexicon would comprise verb-noun couples readable in a two dimensional grid or matrix. Lower, or deeper, layers would be accommodated on parallel planes. Scoping relations, where narrative elements might exist across several layers, being generalised upwards or specified downwards, would link these planes making the whole a three dimensional lattice-like structure. utterances at the action level, indicating transactions should lead to conditional Operations produce changes, facilitated by appropriate switches. This vague conception was never fully articulated.

On the operation level plane, for example, the analyst could cross-reference an appropriate verb, or command, with the appropriate noun, or object, to find a subset of suitable candidate Switches. This was nominally referred to as the operation-switch matrix.

The complexity of natural language however continually produced the need for greater complexity. For example, if a stakeholder said “I get ..” it became necessary to decide if this was a passive doing, suggesting the stakeholder simply received or was sent something; or an active doing, prompting the need to ask if this involved a lookup, find, search, sort, request, download, import etc. In either case, the obvious question follow up would be, “get from whom?” That might necessitate an interview from another party who might indicate some sort of production or dispatch doing.

The notion of assembling sentences to cover all normal stakeholder tasks could not easily be covered by simple noun-verb couplets. At one stage the basic algorithmic building block constructs of computer programs, sequential-action, decision/if and loop/repeat were considered as a grammatical framework.

The error here was becoming apparent. If indeed natural language could be simply and mechanically translated into the terms and grammatical constructs used in computer coding, then the entire need for the research project might collapse. The computer coder should simply be able to code directly from stakeholder interviews. The literature review, however, had established that this could not as yet be done, thus creating the need for structured requirement elicitation, systems analysis and design methods.
Table 4.4: Some Candidate SFA Command Terms

Some of the candidate commands and natural language terms seemed suited to the, as yet unexploited, Rule and Division of Labour elements of AT. As noted in Section 4.15.2 (above) these could possibly be aligned with temporal and deontic logical operators. Terms such as:

**IF, WHEN, UNTIL, WHILE, REPEAT, DURING**

Seemed to offer naturally temporal meaning, whilst others such as,

**ASK, ANSWER, REPLY, ACKNOWLEDGE, MUST, MIGHT, SHOULD, COULD, WOULD, MUSN’T, MIGHT NOT, SHOULDN’T, COULDN’T, WOULDN’T**

Presented a distinctly deontic flavour. Such terms could serve as markers for the presence of these logical operators or constraints, but no clear use for such terms had yet become apparent in the SFA exercise.

The complexity of the three-dimensional lattice SFA lexicon presented considerable and most probably insurmountable difficulties. At the least, it would have been necessary to define a distinct taxonomy of couplet terms to be used at each distinct AT layer; together with clear heuristics for transitioning between layers. Simply to distinguish when a stakeholder utterance has currency one layer or another proved a tangled semantic puzzle. There was simply no way that the SFA notion, however alluring it seemed initially, could yield a simpler AT-theoretic SAD method, not the least because it was drifting too far from the base tenets of AT itself.

### 4.16.3 Temporal and Deontic Operators

The Story Fragment Analysis idea was abandoned because it required more work than could be justified, promised no realistic return and had drifted so far from the original AT-based project that it was no longer within the scope of the original research project.
It was still hoped however, that somehow rendering rules and DivLAB as temporal and deontic operators could either assist in specifying the system or, at the least, in its evaluation, validation or verification.

Lectures and seminars delivered on the research at about this stage of development indicated that Rules were considered to consist fundamentally of ordinal sequencing constraints, and thus should be rendered in terms of temporal logic operators. DivLAB was therefore considered to represent the dependencies and obligations between Activities and thus should be rendered in terms of deontic logical operators. Exactly how either of these translations could occur remained as distant as ever but the hope of potential benefits kept the notion alive.

The relational database model, which still served as a framework for assembling and trialling conceptual developments, was slightly updated with fields to record the temporal and deontic operators best suited to each elicited ‘constraint’ Figure 4.19 shows a sample screen shot of the amended ‘constraint’ record form.

![Figure 4.19: Sample ‘Constraint’ Record Form](image)

Whilst the relational database model now had some nominal place to record temporal and deontic constraints there was little real progress on how best to utilise this information. It was hoped that these constraints or operators would inform the sequential and dependence relationships between doings in the AT network.

Although the SFA diversion had proved ultimately fruitless, efforts to construct quasi-functional descriptive ‘sentences’ had highlighted the relationships between doings. A new importance was placed on the changes and transactions which resulted from operations and actions. The temporal and deontic constraints had the potential to describe and define the nature of these changes and transactions.

### 4.17 When Is A CAT Not A CAT?

Just as SFA had failed, partially because it had drifted too far from AT, it was observed at about this time that the relational database model may also be
questionable, as it indicated far more detail at the action layer than at the activity layer. The concentration on temporal and deontic as constraints upon changes and transactions, likewise seemed to disregard the activity layer.

This situation seriously endangered the basic thrust of the research program, as the AT precepts (Section 2.22.4) dictate that an activity is the unit of analysis, being Motive driven, conducted by a subject with the assistance of some Tools and producing some outcome which might resemble the originally intended Object. Any AT based SAD method must not, therefore cease to concentrate its analysis about the activity.

In Section 4.8, the notion of a Combined Action Table (CAT) was introduced, and in Section 4.11.2 assigned a role as one of four analytical ‘scratchboards’. The realisation that the method under development must refocus its attention at the activity layer raised some serious doubts as to the necessity for a scratchboard based at the Action layer.

The CAT concept seemed too useful and fundamental a ‘scratchboard’ space for analysing doings (sui-generis) to be dropped out of hand, so the CAT was reconceptualised as a Combined Activity Table, and it became a priority to redefine its form and function.

4.18 Returning to AT: Outcome-Tool Exchanges

A refreshed concentration on the core precept of AT, to make the activity the base unit of analysis, required a reconsideration of stakeholder doings at the activity layer. It was important not to fall foul of the temptation to simply abstract, as this would not meaningfully translate the analysis process to the higher layer.

If the units of analysis were to be activities, and if the overall doing of the stakeholder group could be conceived as a network of such activities, then some useful details must surely reside in the linkages and relationships between these nodes. If, at the action layer, there are transactions, then perhaps at the activity layer there would also be some form of exchange. But, what might one activity send to, or receive from, another? What are the defining characteristics of the things transferred between activities?

Figure 4.7, the 5S concept kidney diagram graphically presented the early realisation that the interface elements of the system necessarily resided at its boundaries. It was probable that some of the things exchanged between the activities of the final system might be transferred internally, within and between the somewhat automated ‘mechanisms’ of the system itself. Those activities in which a human user interfaced with the system however should get, give or otherwise interact with some thing across the system boundary. Such exchanges, and the activity doings which produce, receive, consume or create them, must therefore be crucial to the interface design.

Engström’s AT matrix indicates that an activity produces some sort of output, or has some form of outcome, which may or may not quite meet or satisfy the motivating original object. It is also noted in AT that the outcome of one activity may be received into another activity as almost any element. One activity may, for example,
produce a rule under which another operates, or provide some tool which another uses.

Since this project’s interpretation of AT was deliberately tool-centric, choosing to describe a system as a set of agenda-facilitating computer-based tools, it seemed logical to consider those relationships in which the outcome of one activity provided a tool to another.

The focus of research therefore became, for a time, those outcome-tool exchanges which passed across the system boundary. It became necessary to investigate, describe and define those *things* which passed between activities in such exchanges.

4.18.1 Instruments

The earlier concentration on transactions (see Figure 4.18) which was associated with the notion of story fragment analysis (see section 4.16.1 to 4.16.3), and lead to the focus on outcome-tool exchanges, (above) was to prove a fruitful diversion. If a complex agenda conducted by a group consists of a network of activities linked by transactional exchanges, then the need arises to identify, describe and classify those *things* which might be passed between activities in such exchanges.

It is important to note that, in the conceptual stages of system design, the system itself might be imagined as an incompletely described computer-based entity (or group of such entities) residing in some space *between* the activities of the involved stakeholders (users). The more activities have this system-actor as their subject, the higher the degree of automation across the group’s process. Usability and acceptance of the system by the group is sensitive to the degree and implementation of automation. The method under construction, being thoroughly rooted in an analysis of the users’ doings shows potential to improve such design decisions.

The boundary membrane of the system space would contain the elements of the human-system interface(s), most probably located at points where the system space is somewhat tangential to an involved activity. Each and every exchanged *thing* which might pass between a user and the system (or to another user via the system) would pierce and pass through the system boundary and, depending on the degree of automation involved, is likely to involve some aspect or element of a human-system interface. This notion is entirely consistent with the early ‘5S’ conceptual description of the product of the method under production (see section 4.11.1 and Figure 4.7).

The overall objective of the research project, to explore the possibility of an activity theory based software system analysis and design method. Accordingly, only those transacted *things* which might be passed through, created by, stored within, modified using or consumed by a software system are of specific interest. Transacted *things* which might not directly touch the system are of lesser concern to this project, so long as any contextual information or constraints they may infer, imply or embody are considered and recorded in the agenda, motivations, goals or conditions of the system analysis and design.

The term Instrument was adopted to describe all those transacted things which pass between activities and cross the system boundary; thus appearing in some form within the system itself. The earlier and seemingly similar term ‘artefact’ adopted by
McGrath (2003) was not used because he had a markedly entirely different conception (combining both tools and objectives). Instruments may include such divers entities as files, data, time stamps, logons, log entries, permissions or any other measurable and transmissible ‘event’ within or between a software systems.

AT scholars (notably Kaptelinin 1992a, 1992b, 1996) indicate that a number of notable linkages may exist between activities; that is, that an activity’s outcomes may yield more than just tools for other activities. The method currently under construction however, being concerned with the nature of the tool(s) which represent a facilitating computer system and its interface(s), was predicated on a tool-centricity. This method was, therefore, concerned almost exclusively with outcome-tool transactions, and in this latest conception, chiefly those which *might* pass through an, as yet undescribed, facilitating computer system.

It is important to observe that instruments exist, and are transacted, at the Activity level. (see section 4.10.1 and Table 4.1) The deviations from the base precepts of AT tended to involve at least some descent into action or even operation levels, and yet the Instrument concept operates most logically at the activity level, consistent with the fundamental AT maxim, that the activity is the base unit of analysis (Vygotsky 1978).

It was realised that any functional group operating to an agenda, will be exchanging *Instrument-like* objects even before the design or implementation of any bespoke computer based system. Though no single customised system may yet exist, such a group will already use numerous tools, including documents, techniques and even some computerised tools. Numerous data-like forms, records, lists, dates, requests, receipts, templates, flags and permissions would be generated during operation and it is likely that a number of these will have been ‘standardised’ in some material form, most likely as paper documents and/or stored or otherwise manipulated using extant computer-based tools. Any or all of these might be considered as pre-cursor, proto or candidate Instruments.

An early stage of elicitation and analysis should, therefore, involve the identification and description of such candidate Instruments, as these will likely form a useful basis for describing the final set of computerised instruments. If any newly designed system is to be implemented with the least disruption to the group process, its instruments might do well to imitate, reproduce or otherwise respect the proto-instruments used in the previous processes – this would be consistent with the widely accepted tenets of good interface design such as those indicated by Nielson’s famous heuristics (Nielson 1994).

Figure 4.20 and Figure 4.21 show how the instrument notion was added to the relational database model, which continued to be a useful place to record the evolving form of the SAD method under construction. Observe that rule and DivLAB elements are still deemed best described as temporal and deontic constraints, but could now be entered in their own table and recorded against specific Instruments. It was considered that these constraints would impact upon the exchange of instruments between activities.
These simple database additions merely provided a repository for instruments to be recorded, they failed to clearly illustrate the central role of the instrument. Instrument transactions form the links between activity ‘nodes’ at the network level. Instruments encapsulate the interesting outcomes from activities and their passage through the System boundary correlates directly with the occurrence of human-system interface elements.

For the purposes of understanding and designing a highly interactive multi-user computer system and its interface, the doings within and among the group seem to be describable largely in terms of the Instruments exchanged. If activities are the principle organs of the group body, then instruments are their lifeblood. A thorough study of instruments and their functions was required.

4.19 The Penny Drops - From UI Design To SAD

The method under construction had, at one time, been motivated largely towards generating designs for a system’s UI. Any interaction or transaction resulting from a user’s activity, which involves the system in any way, yields an instrument. All significant doings involve the creation, reception and/or modification of one or more instruments, each of which passes through the system boundary (between the system and one or more of its users) at least once. By definition then, UI elements exist at those points where instruments pass through the system boundary. As indicated in Figure 4.22, those activities, whose tools include instruments used by the system, may be considered to be located tangentially to the system boundary. The UI involved in the doing of any specific activity, consists of widgets that interact with the instruments listed for that activity.

In order to define the UI, it became necessary to be able to enumerate and describe all instruments. This should yield a listing of all required UI elements, some notion of their required natures and functions, and (perhaps most importantly) their clustering, or grouping, into specific ‘screens’ according to their user’s role. A listing of Instruments must, therefore, encompass an understanding and description of all significant Activities (and sub-doings) together with their relative dependencies (both temporal and deontic) and contain descriptions of all data generated, stored, transformed, translated, retrieved and consumed.
It quickly became apparent that if such information could be elicited, analysed and expressed, it would describe not only the operational interface elements of a given system, but provide a grey-box specification of its functions and operations. A sufficiently detailed and complete AT-oriented instrument-based description of a group’s process, designed around a proposed bespoke facilitating computer system, must necessarily describe both the form and nature of its UI and also the transactions, translations and transformations of data within it. Given a sufficiently detailed account of where data comes from, how it is manipulated and where it must go to, one is able to describe the system itself.

It was realised that the method under construction, ‘5S’, if it could be made to function, promised not just a UI design method but a complete SAD method. No known SAD, at the time of this writing contained within it an integrated descriptor for UI specification. Neither could any known UI design method claim SAD capability. Indeed, these two dimensions seem, in the literature, to have remained distinct and at times, even disciplinarily incommensurable. If it could be shown to be at all workable, it seemed the ‘5S’ method might promise some potential improvement over extant methods. It remains beyond the scope of this current project however to critically evaluate the performance of ‘5S’ against extant SAD (or UI design) methods.
If the method under construction could produce feasible specifications for both UI and for the system, starting from the earliest phases of stakeholder elicitation, then it would have currency across an unusually large proportion of a software project’s development life-cycle. If AT could provide an end-phase mechanism for validating specifications, verifying that they were met, and evaluating the final product (by some means perhaps similar to Vrazalic’s Distributed Usability Evaluation Method (DUEM) (Vrazalic 2004), possibly combined with elements of the SFA notion explored during this chapter (section 4.16.1)) then there may be grounds for proposing a genuine end-to-end methodology. At this time, such breadth lies outside the scope of the research project, but the possibility flags future research directions.

4.20 The Unexpected Persistence of Activity Theory

It was noted, at this point in the construction of the method, that despite the tendency for speculative method building to wander, the fundamental precepts of AT repeatedly re-asserted themselves. It proved necessary to return to the basic precepts of AT in section 4.14.1 (above) after becoming entangled in the minutiae of defining user micro actions, again when the CAT was redefined in section 4.17 (above) and once more in section 4.18 (above).

Whilst the broad character of AT suggests that it ought to suit UI design (Bødker 1991), and the possibility that it might suit SAD has been noted (Mwanza 2002), to date creating a prescriptive AT based method had proven elusive. As observed in chapter 2, Leon’tiev’s sliding hierarchy proved both a powerful conceptual framework and a monumental source of ambiguity. Most attempts at an AT-based design method had failed to cope with the inherent vagueness of AT. Even the sophisticated work of the Vrazalic thesis had been unable to clearly identify which elements of a stakeholder’s agenda were instantiated as Actions or as Activities. Most attempts to date have required an additional layer of theory or practice, such as Distributed Usability, Articulation Theory or Unified Modelling Language (UML) activity diagrams and still fallen short of exploiting AT’s full potential.

Despite recognition of the highest, activity network layer and the adoption of a standardised hierarchic taxonomy, this research project too, tended to lose its way in Leon’tiev’s labyrinthine hierarchy. The construction process detailed in this chapter
reflects the ambiguous and problematic nature of adapting AT into prescriptive practices.

As was the case with several previously published method building attempts (Fjuk 1997, Mwanza 2001 & 2002, Fjeld 2002, and Vrazalic 2004), this method started to stray away from AT and the addition of further theories and frameworks was considered, largely to assist in tying AT’s slippery framework down long enough to make use of its promised power. Ultimately, AT proved itself remarkably resilient. In the construction of this project’s method, the best way to resolve or remediate confusions repeatedly proved to be a simple return to the most basic AT principles.

Whilst at this point, construction of the method was by no means complete, AT was proving surprisingly persistent. There were grounds to feel confident that the initial hypothesis emerging from the literature review and methodology discussions had validity. AT seemed indeed capable of serving as the basis for a UI and SAD method.

As discussed in chapter 3, normative validation of the hypothesis would depend upon demonstrating that a workable AT-based method could be constructed. It would be necessary to return to the trial case and re-examine it from the newly reinforced Activity-layer notion of Instrument transactions. Initially, it would prove necessary consider what changes to the method conception may be entailed by recognising that it is nearer to being a complete SAD method than just a UI design method.

4.20.1 Cooking the CAT

With the commitment to the basic precepts of AT reaffirmed, it was necessary to re-examine the practicalities of the methods’ workflow. A fresh focus on how the method might apply against realistic situations could refine its mechanisms.

The somewhat idealistic fourth workflow concept represented by the seven phased U diagram (Figure 4.10, section 4.13) offered a single deconstruction pass from activity network down to atomic operations. After somehow mapping operational conditions to UI component switches, a single recomposition pass should assemble the specifications of the system. Many of the difficulties and tangential efforts described in preceding sections arose during explorations into how this pivotal mapping of operations to switches might be achieved.

The U diagram workflow dictates that CAT should emerge at the end of the third phase, even though that phase concentrates on actions rather than the more abstract Activities. The ambiguities and confusions stemming from Leont’ev’s sliding AT scale were thrown into sharp relief when the CAT was redefined from Combined Action Table, to Combined Activity Table, cementing its place at the activity level.

In section 4.19, the Method under construction was found to show considerable potential for SAD. The very term infers two distinct stages, analysis and design. Consistent with this project’s adopted notion of design (section 4.3), it is necessary to ascertain stakeholder requirements before attempting to satisfying them. This makes it necessary to run an analysis pass to identify requirements, then a design pass to reconfigure or rearrange elements into something more desirable or effective, prior to any system being constructed.
A stakeholder group with some need for a facilitating multi-user computerised tool will necessarily already have some form of systematic process (even if it only exists as a concept or notion) however inefficient, *ad hoc* or filled with redundant doings. It is given that the stakeholder group perceives some potential for improvement or else they would not commission the design of a new or improved tool. The initial elicitation feeds into a consensually agreed picture of the processes the stakeholders *currently* use. This is followed by the design pass, more properly a *redesign* pass, to reach a consensually agreed model of a better process. If described systematically, the agreed model of a better process can serve as the requirement specification for a new computer system.

Consensually negotiated maps of business doings can disambiguate differing impressions of group processes granting each group member an opportunity to see their own doings as part of a greater whole. The value of such process maps has been noted in the literature and a recognisable sub discipline of Business Process Modelling (BPM) exists between commercial management and Requirement Engineering (RE), devising mapping techniques and defining new notations, taxonomies and semantics. One such hybrid toolset, the *i*\* (*eye-star*) mark-up notation serves both as a BPM tool and part of a larger RE effort to reason with so-called Non-Functional Requirements (NFR’s) (Chung *et al.* 2000).

Should the AT-based SAD/UI method under construction yield a feasible AT-based BPM method, that would be a worthwhile outcome in, and of, itself. Potential BPM applications of this method must be flagged for potential future investigation.

The fourth workflow concept was amended to accommodate the notion that final design must follow analysis. Initially it was decided that the U diagram workflow could provide analysis and a third pass would provide design. Having, supposedly, elicited all necessary analytical information from decomposition and recomposition passes down and up the Leont’evian hierarchy, it was imagined that a system specification could be designed along similar lines, pass from the abstract to the detailed. This third pass would; gather role data about users and their organisation relations, gather instrument data about the ‘pseudo-data’ and meta-data they would operate with and upon, build a data dictionary defining the variables and flags the system would manipulate, and finally provide a full data flow specification that could be provided directly to software coders. This minimally amended workflow is illustrated in Figure 4.23 the Amended U Diagram.

The amended fourth workflow, however, had not resolved any method by which operational conditions could be mapped to interface switches, nor had it addressed the persistent confusing ambiguities of Leont’ev’s sliding hierarchy. Reconsideration of the trial case produced a realisation that whilst the description of the stakeholders’ extant ‘system’ may well proceed decompositionally downward through Leont’ev’s hierarchy, actual ‘design’ consists of a re-imagining of the stakeholder’s processes and doings. Such a re-imagining would comprise a ‘decomposition’ of a proposed new system, and as such, would also follow a downwards path through the hierarchy. The generation of usable system specification for coders to work with would, necessarily, result ultimately in a broad description, inferring an upward path towards abstraction.
Figure 4.23: The ‘Amended U diagram’

Since stakeholders would not normally be expected to possess a complete, accurate or AT-oriented BPM of their extant ‘system’ (if indeed, any at all) the very act of eliciting and describing their current processes (a consensually negotiative process) could be expected to throw up suggestions and ideas for streamlining, automation and refinement. This realisation meant that the initial decompositional path would probably run parallel and just a little before the refinement (or proposed design) pass.

Figure 4.24 illustrates the fifth workflow concept which emerged from these considerations. It featured double, though not quite simultaneous, downward passes producing the BPM of the extant system and the proposed refinements. Final consolidation of the specification would occur as a distinct upward, recompositional, pass. This concept was dubbed the W Diagram, as the figure itself might suggest.

Although the re-design was fed by suggestions and observed inadequacies emerging from the initial BPM, it was important to note that one simply cannot simultaneously re-design a thing that is not yet known. By recognising that, whilst linked, the description of extant processes and the investigation/design of refined processes were separate components of the method workflow, considerably less complexity was required of the method at any one point.

4.21 The Trial Case

Although the fifth workflow concept, illustrated in Figure 4.24’s W diagram, promised a more realistic notion of the method, it still contained unresolved problems. Most notable was the mechanism by which operations and their motivating conditions might lead the analyst/designer to select appropriate interface Switches. It was decided to return to the ‘Academic Assessment’ trial case to select a small subset of doings and pursue how analysis and redesign might proceed.
It must be acknowledged that details in this trial case were not elicited from any real-world stakeholders, but extracted from the author’s extensive personal history in designing, conducting, marking and administering academic assessment tasks. This section, then, does not test the initial elicitation nor the negotiation of consensually agreed BPM. This exercise serves only to explore specific mechanisms within the method under construction. Some results and discussion of this section have been reported in a peer-reviewed Journal publication (Brown et al. 2006c).

In this subset of the process, an Academic designs an assessment tasks for Students to complete. The informing agenda is that: ‘Students must demonstrate their learning & skills by completing an indicative assessment task to a measurable predetermined standard without cheating’.

Deploying domain specific theory obtained externally to this process, the Academic creates both a set of questions, and a set of answers which serve as a marking guide. The Students obtain a copy of the assessment questions and, applying knowledge received in earlier classes (external to this process), based on the same theory from which the assessment questions are drawn, create and submit answers. The Academic collects and marks these answers, according to the correctness of the Students’ application of the received theory. The Academic must collate and register all these mark results with some external administration system.

To simplify the trial case and concentrate on a representative subset of doings, the rôle of Tutor, or Teachers-Assistant, who collects and marks student assignments on behalf of the Academic, has been removed as has some specific actions associated with verification of student enrolment and identity, the appealing of mark results and other largely peripheral processes. Only the central elements of the academic assessment process will be considered in this section; which is, to write questions,
write a marking guide, send questions, answer questions, send answers and mark answers.

### 4.21.1 Adjacency Matrix

Only two user types, or subjects, are considered: S1 the ‘Academic’ and S2 the ‘Students’. The data-like entities initially considered as candidate instruments are the Questions (Q), the Marking Guide (MG) and the Answers (A). Some other Instrument-like entities, which are initially considered to be external to the system, are informing domain specific theory; and the final report which is ultimately sent to some external administration system.

The doings listed below in Table 4.5, which most probably exist at the Activity level, are represented in the Activity Network, illustrated in Figure 4.24:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Subject</th>
<th>Motive</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1.1</td>
<td>Academic</td>
<td>Create a question document (Q) based on theory</td>
</tr>
<tr>
<td>S1.2</td>
<td>Academic</td>
<td>Send the question document to the students</td>
</tr>
<tr>
<td>S1.3</td>
<td>Academic</td>
<td>Create a marking guide (MG)</td>
</tr>
<tr>
<td>S1.4</td>
<td>Academic</td>
<td>Get the submitted answers (A)</td>
</tr>
<tr>
<td>S1.5</td>
<td>Academic</td>
<td>Generate a report of how answers rated against the marking guide</td>
</tr>
<tr>
<td>S2.1</td>
<td>Students</td>
<td>Get the Question document</td>
</tr>
<tr>
<td>S2.2</td>
<td>Students</td>
<td>Using the Questions, create an Answer document</td>
</tr>
<tr>
<td>S2.3</td>
<td>Students</td>
<td>Submit the Answers, and Modify as necessary, prior to deadline</td>
</tr>
</tbody>
</table>

**Table 4.5: Initial Doings, Candidate Activities**

The initial lists of doings, given above, are laid out in a simple directed-graph form to illustrate the Activity Network. Figure 4.25 shows the simple directional passage of the candidate Instruments Q, MG and A between the listed ‘doings’ of the subjects S1 (Academic) and S2 (Students). In this simple case there is no obvious ambiguity as to the sequencing or interdependence of these doings. This simplicity results from both the selection of a deliberately reduced set of doings, and the fact that the details were produced purely from the author’s personal experience. Details elicited from a diverse group of stakeholders may be expected to contain inconsistencies. As with recognised BPM mechanisms, the very act of attempting to draw up such a list of doings and directed-graph should assist in allowing each stakeholder member (and the analyst) an abstracted overview in consistent terms. This should aid in negotiating consensual agreement between stakeholder members as to their interpretation of their extant processes and doings.

It is a relatively simple matter to render a directed graph into an adjacency matrix as shown in Table 4.6. Such a matrix simply presents the transacted entities in a tabular form, against those nodes which send and receive them. This adjacency matrix forms the CAT. It becomes clear then, that the CAT is no more than a convenient representation of the directed-graph form of an activity network.
If the stakeholders have achieved consensus on this representation of their process, then it becomes possible to consider suggestions and reactions to possible inefficiencies or redundancies within it. Numerous suggested ‘improvements’ may actually arise during initial elicitation and negotiation. The analyst should note these for consideration in the second downward pass of the workflow. This partial simultaneity between initial analysis and refinement is represented by the connective arrows between the two downward passes of the fifth workflow concept, illustrated by Figure 4.24.

It is possible that some doings and transitions may initially present as redundant or inefficient for several reasons. Firstly, they may genuinely be redundant, producing an un-needed Instrument or sending it to a doing which does not require it. Secondly they may genuinely be inefficient, operating on, producing or consuming an instrument which might be better used in some other part of the network. Thirdly, the individual(s) suggesting that a doing or transition is redundant or inefficient may not understand its purpose. It is one of the goals of a consensually agreed BPM that such misunderstandings can be addressed to assist each stakeholder to grasp an abstracted overview of the broader organisational context. Such mediations require the BPM be rendered in a mutually comprehensible and disambiguated taxonomy and language. Finally, if the driving condition, goal or motive of each doing and transaction (or the agenda of the entire activity network) has not been considered, then the purpose of each may not be clear to all parties. This lack of information could lead to elements being incorrectly marked as redundant or inefficient. It is hoped that AT’s focus on such informing motivators will carry through the method under construction and help to remove misinformed false classifications. Though difficult and even somewhat clumsy to record in any formal or discretely mathematical manner, driving motivators should be elicited and recorded by the Analyst and should form an important part of any elicitation process. Driving motivations provide important contextual markers, useful in informing later design decisions.
Table 4.6: The CAT Adjacency Matrix for the Activity Network in Figure 4.25

<table>
<thead>
<tr>
<th>To From</th>
<th>S1.1</th>
<th>S1.2</th>
<th>S1.3</th>
<th>S1.4</th>
<th>S1.5</th>
<th>S2.1</th>
<th>S2.2</th>
<th>S2.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1.1</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1.2</td>
<td>Q</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1.3</td>
<td>Q</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1.4</td>
<td></td>
<td></td>
<td>-</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1.5</td>
<td></td>
<td>MG</td>
<td>A</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S2.1</td>
<td></td>
<td>Q</td>
<td></td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S2.2</td>
<td></td>
<td></td>
<td>Q</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S2.3</td>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Assuming that a consensual CAT is produced and driving motivators (context) have been agreed and understood by all parties, it becomes necessary to consider refinement of the process. Initially redundancy and inefficiency should be identified, and then mechanisms whereby simplification can be achieved by redirecting transactions through the agency of the system under design (which does not yet exist).

It may initially seem contra-indicated to assign function to an, as yet non-existent system entity, but the reader is invited to recall the informing concept of this project; that the system may be described by the functions, roles and doings which operate at its periphery. The analyst is expected to imagine a central facilitating system which will resides between the doings near the centre of the activity network. It is hoped that a feasibly improved refinement of the group’s process may result from arranging their doings and transactions such that as many as possible sit close to the system boundary and transact their Instruments through it. The system itself will be the most important new tool common to those activities which it supports. Describing those refined and re-ordered activities will, it is hoped, describe the system that supports them.

A new difficulty arose at this stage of working through the trial case. It was unclear if refinement should concentrate on the directed-graph nodes (the doings) or upon its arcs (Instrument transactions). Each transaction necessarily involves two doings, one which produces the transacted instrument, and one which needs to consume that Instrument. Each doing’s driving motivation to transact an Instrument necessarily involves the product or need of some other doing, and no transaction is adequately described without consideration of both. This chicken-before-the-egg conundrum had its origins in the attempt to describe Entity Relationship (ER) tables for a supporting relational database tool (Section 4.15). It became necessary to take a fresh look at the issue and to attempt to describe new heuristics for refining the activity network.

4.21.2 Heuristics for Graph Pruning

The activity network and its associated CAT form a primitive BPM which facilitates an abstract consideration of the overall process. Examination of these representations should reveal redundancies and inefficiencies which can be addressed in a later design pass.
Chapter 4 – Method Building

Consistent with the fifth workflow concept illustrated by Figure 4.24, it is appropriate to consider how inefficiencies and redundancies might be addressed and refined. The ultimate goal is to facilitate a refined process and reduce the human workload, by automating as many Instrument transactions as possible through an, as yet non-existent, system. In effect, the system will (once created) form a central and principle subject (do-er) which participates in all Instrument transactions. Recall that, in the final design, all instruments (by the definition offered in section 4.18.1) will cross the system-user boundary in some way.

To form refining heuristics, it is best to seek to simplify the directed graph. The Analyst should attempt to eliminate ‘unnecessary plumbing’ from the graph. Two types of unnecessary plumbing are considered: Pipes – which are those arcs (instrument transactions) between activities which have a common subject (owner) whereby the Subject sends something to themselves; and Joints which are nodes (activities) through which an Instrument passes unchanged.

In Figure 4.256’s simple directed graph, Pipes run between the following activities: S1.1 to S1.2, S1.1 to S1.3, S1.3 to S1.5, S1.5 to S1.4, S2.1 to S2.2 and S2.2 to S2.3. The joint nodes are: S1.2, S1.4, S2.1 and S2.3. Figure 4.25 indicates the pipes by shaded ovals overlaying arcs, and joints with dashed circles.

It is considered that the strongest candidates for collapse, removal or merger are those activities which are both a joint and reside at one end of a pipe. Thus, activity S1.2 “Academic sends Q” is a strong candidate.

Following this heuristic, it is possible to collapsing the pipe S1.1 to S1.2 by absorbing S1.2 into S1.1. Arbitrarily, a heuristic is adopted which assigns the numerically lowest of two activity numbers to identify a newly formed combined node.

Likewise, activity S1.4 “Academic gets A” is combined with S1.5 and retains the lower number, S1.4. The resulting graph is illustrated in Figure 4.27.
Considering the subset of activities which involve the Academic (S1), a weaker case for possible collapse exists between nodes S1.1, S1.3 and S1.4. Three candidate refinements seem possible, each collapsing a different set of activities.

**Figure 4.27: Partially Refined Activity Network**

The first option is to collapse all three of these activities into one. There is a clear temporal issue at work here. It will not be possible to compare a student’s submitted answer (an instance of A, which is their response to the assigned Questions (Q)) against the Marking Guide (MG), before Q and MG have been created. Only by automating the student’s creation of an answer, such that the entire process runs in near zero time is this possible. Given that the driving agenda of this entire activity network is specifically to assess the student’s ability to answer the questions, the creation of Q and of MG simply cannot occur at the same time as the comparison of A with MG. It is important to note that a simple consideration of the driving motivation, whose importance is highlighted by the precepts of AT itself, resolves this option and this candidate for collapse is rejected.

Secondly, it initially seems possible to collapse S1.3 and S1.4, simultaneously creating the marking guide, marking A and generating a report. Though this trial case is driven by the author’s own experience, it is reasonable to assume that a stakeholder group would quickly point out that whilst there is one MG document, there are many instances of A, one submitted from each student. This one-to-many temporal relationship indicates that the single instance of MG must serve multiple ‘marking sessions’. If correctly elicited, the rules and DivLAB should have made this limitation clear. This candidate for collapse is also rejected.

Finally, it may be possible to collapse S1.1 and S1.3. Creating both the questions and the marking guide in one overall doing seems a natural pairing, as both are products of the domain specific theory and its prior delivery to the Students through classroom lessons. As no temporal or other restrictions are immediately apparent, activity S1.3 is collapsed into S1.1.
Returning to consider those activities conducted by Students (S2), Figure 4.27 indicates that both S2.1 and S2.3 are strong candidates for collapse into S2.2. The driving motivation of the Process, its agenda, however has been recorded in section 4.21 as requiring that: ‘Students must demonstrate their learning & skills by completing an indicative assessment task to a measurable predetermined standard without cheating’. This is, to test if students can successfully convert Q into an acceptable A (according to the MG) by way of applying their previously received lessons. From this, an analyst may see that the whole purpose of these student doings is for them to conduct personal creation of an answer set from a question set. Doing S2.1 receives Q and doing S2.3 submits their A.

If S2.1 and S2.3 are allowed to collapse into S2.2, then the process of creating an A from a Q becomes automated. The entire process quickly collapses into a trivial case, whereby the student is simply assigned a 100% mark. The student may be removed from the process. At a larger scale, where there is some administrative system (beyond the scope of our activity Network) which sends enrolment data to our assessment system, and receives a marking report in return, then a student would be automatically awarded top marks as soon as they enrol in the subject. The entire process may be reduced to a trivial mapping within the external administration process. The S2 candidates for collapse must be rejected.

This demonstrates quite clearly the need to record and consider Driving Motivations, as the contextual information they convey is necessary to prevent catastrophic design decisions. At this time, such considerations would require the agency and intervention of a well-informed human analyst-designer. Further automation of the analysis-design procedure would require codification of the driving motivations and the constraints they entail. Though it is beyond the scope of the current study, there is considerable value in future research towards recording motivational constraints as more formalised and discretely mathematical temporal (sequential) and deontic (dependence) clauses. When expressed under a multi-valued logic, a computer facilitated constraint satisfaction analysis could greatly enhance the analysis-design process.

Observe that activity S2.2 “Student uses the Questions to create an Answer document” must occur outside of the system under design. In fact, the processing of S2.2 must necessarily occur within the students’ own brain. The candidate Instruments passed between S2.1 and S2.2, and between S2.2 and S2.3 cannot therefore be ‘computerised documents’ since we currently lack the technology to directly interface such objects with the human brain. The Student must instead read Q into their personal brain-system, and write A from their personal brain-system using more traditional physical and psychological tools.

If an outcome-tool transaction between activities does not pass through the system, then the data object passed by that transaction is not an instrument, by the method’s definition. For clarity and to retain a sense of context, these objects should remain recorded in our mapping. The data objects are handled, used and created inside the Students’ personal brain-system are non-instrument interpretational variants of instruments Q and A. These are recorded as Q* and A*. As there is no longer an unchanged Instrument passing through doings S2.1 and S2.3, their classification as joints lapses. The case for collapse of the S2 Activities has been further weakened.
and remains rejected. Activities S2.1 and S2.3 should now be considered to have translational and transformational qualities, interfacing with an external system; in this case the Students’ personal brain-system.

Figure 4.28 shows how these remaining activities relate. The instruments Q, A and MG reside in the System Data Repository (SDR), whilst the non-instrument objects A*, Q*, Theory* and Report* do not. An early depiction of the system emerges at this point, tentatively defined by the instruments it handles, their natures and the temporal and deontic constraints applying to their transactions.

After applying these pruning heuristics the stakeholders might be consulted to reach mutual satisfaction with the design. The activities at this stage include: S1.1 (create and modify a Q document), S1.6 (create and modify an instance of MG) and S1.4 (create and modify marks for submitted student answers & create a final report). The non-sequential numbering of these Activities has resulted from the refinement and design process detailed above.

Peripheral non-System Activities, retained for context, include: S2.1 (Get the Q document), S2.2 (Using the Qs, create an A document (applying received knowledge)) and S2.3 (Submit the As, and modify as necessary, prior to deadline).

Lists of this kind exists when the analyst-designer is approximately at the bottom of the ‘refinement’ pass given in the 5S workflow W diagram (see Figure 4.24) and is about to undertake the ‘recomposition’ pass.

Figure 4.29 shows activities and instruments after the designer has rationalised the transactions. The identified instruments are: the question document (Q), the marking guide (MG), the Students’ answers (A) and the marks assigned by the Academic (Mks).
Important points arising from this exercise were: Firstly that the driving motivators are very important, as are DivLAB and rules, because they constrain blind application of simple pruning heuristics. They direct the analyst’s elicitation, guiding negotiation towards important issues. It is important reasons to capture and record these issues in a disambiguated manner. There is scope for them to be formalised as Temporal and Deontic constraint clauses.

Secondly, initial elicitation and refinement may be conducted almost simultaneously. This reinforces the incremental joins shown between these two passes as illustrated in the fifth workflow concept W diagram (see Figure 4.24). These passes yield indicators towards recomposition (design). As with the earlier passes, recomposition must also be sensitive to guidance from the driving motivations, especially high level ones such as the group agenda and activity motives.

At this time, it was anticipated that recomposition (design) would likely involve numerous modifications to lower level doings, such as actions and operations. Some would be combined, deleted, created or rewritten. Some may even be promoted or demoted to higher or lower levels. Constraints initially recorded against Actions (goals) and operations (conditions) may perhaps be open to relaxation, and be a little less rigorously applied than the constraints entailed in the group’s agenda and activity motives.

### 4.21.3 Recomposition / Design

The designer now turns to the elicited constraints and motivations, to maintain the user-orientation which underlies this entire project, should also consult with the stakeholder group. Consultations yield modified drivers and rationales which shape new doings. Some initially identified doings may be automated (subsumed into the system entirely), some may be deleted; others may be merged or split. Some entirely new activities or actions may be defined as deeper exploration reveals doings not previously elicited, or in response to newly designed pathways of instrument transaction. Likewise, any number of new or modified Instruments may be described.

For this exploratory trial case such consultations are simulated by drawing upon the academic teaching experiences of the author and also of Informatics Faculty members of the University of Wollongong (Australia) through casual conversation.
From these, the recomposition / design process may be simulated in order to extend and refine the method under construction.

Under consultation, S1 (Academic) indicates a desire to modify MG, even up to the point where early answers (A) are being received from S2. This is temporally separate from the creation of Q, so we create a new activity (with a new number) S1.6 “Academic modifies MG” which reads Q, A & MG and writes to MG.

S1 further indicates that they do not create a Report* for each student, but one summary report. Therefore it is necessary to accumulate the scores for each A, until a Report* can be finalised. We therefore create a new instrument Mkς (marks) which is written to and modified by S1.4 It is apparent that the Report* cannot be generated until all answers have been marked, so we need a flag-type instrument “Mkς-Done”. It is possible for us to automate the setting of Mkς-Done to the value TRUE when the number of received A’s equals the number of members of S2 (Students). S1 however does not want this to happen as soon as the final student submits, so we make the setting of Mkς-Done a manual user operation. An instrument which holds the number of members of S2 could be created, but S1 doesn’t deem it necessary as the value may simply be imported from an external campus level administration system.

S1 expresses a desire to modify the Q document, and not release it to the students until he/she is satisfied. We create a manually set instrument “Q-Done”, which must be set TRUE before Q can be sent to the students. For the students to be aware that Q is ready to be collected there can be a number of further instruments, but for clarity, these are omitted here.

S2 (Students) indicate that they wish to be able to correct and re-submit answers. This means that Activity S2.3 becomes “submit and modify A”. Experience with the modifications of Q and of MG suggests we could create a user-settable “A-done” instrument could be created. The Academic (S1) however, who has higher authority over the design, vetoes allowing S2 this degree of control and opts for “A-Done” to be set TRUE when a pre-set deadline expires. From this, it can be easily deduced that S1 must earlier set a value for a new instrument “DueDate”.

In previous publications [Brown et al. 2006a, 2006b] the activity whereby S2 ‘gets’ Q was decomposed in detail. The potential for numerous new instruments was observed here, to facilitate validation of the student, their selection of the correct subject and assessment task, and the mode of by which Q is received. For clarity, these were omitted in this purely exploratory trial case.

Table 4.7 presents the Instrument Table (listed by activity) for the designed system after these consultative design decisions. For each activity we list its associated instruments. Each activity has an operational relationship with its associated Instruments such as: read (R) the data or status of the instrument, read and/or write (R/W) the data or status of the instrument, setting (Set) the status of a flag, requiring (Req) a particular condition to be true of the Instrument, or creating (Create) an entirely new instance of an Instrument, typically for external use. An asterisk indicates the non-instrument objects passed by non-system transactions which are here included for context.
The Instrument Table, which details the objects handled by the SDR with their constraints and dependencies, could serve as the basis for a traditional Data Dictionary or similar specification document. In producing an instrument table and a hierarchic activity network analysis, the method under construction continues to show the potential for both full specification of the UI and of the system requirements.

4.22 Consequences of the Trial Case

In section 4.21 the method was investigated and enhanced through applying its concepts against a realistic but purely artificial trial case. Whilst the theoretical components of the method seemed sensible and appropriate, under trial application opportunities for refinement were identified. It is necessary to review and consider the consequences of running through this trial case.

4.22.1 Review the Workflow

Members of the stakeholder Group are unlikely to have a working knowledge of Activity Theory, let alone the 5S method, and so are unlikely to have an Activity oriented view of their business processes.

The trial case highlighted that the previously proposed sequence of elicitation questions (section 4.14.2) and workflows did not closely match the information stakeholders would typically be able to deliver. As predicted by an anonymously hostile commentator reviewing a draft for an early publication on this research programme, it indeed proved unlikely that a user can simply utter specifications (not that this project ever lodged any such claim).

The workflow conceptions described thus far have predominantly obeyed the implicit sequencing of Leon’tev’s hierarchy of activity, action and operation. Even
the fifth workflow concept (Figure 4.24, the W diagram) contains an assumption that it is necessary to decompose from the abstract to the refined according as described in Leon’tev’s hierarchy. Though never stated, the diagrammatic workflow representations also suggest an equivalent effort or productivity for each sequential segment, which was neither predicted in AT, nor found in the Trial case.

The proposed phase one elicitation questions (Table 4.3), were an early indicator that users may not naturally describe their doings in exactly this sequence. If typical stakeholder groups viewed, and could describe, their processes in this way there would likely be markedly less need for the discipline of BPM and for the process-oriented mappings required by the ISO:9001 Quality Management standard (ISO:9000 2008). A consequence of the trial case therefore, is recognition of the need to seek first whatever information is initially available from a typical stakeholder group. The method must be free to elicit and assemble data from wherever it is available, regardless of any nominal sequencing, or timing, inferred by Leon’tev’s hierarchy.

The stakeholder group may not have a consensually agreed, unambiguous activity-oriented view of its process, but is most likely to know its members and their relative ranking within the group. It is reasonable to expect that businesses will have some form of an Organisation Chart, or could produce one with relative ease.

Modern businesses, especially with an eye to ISO9001 compliance, may have compiled detailed Position Descriptions (PDs) of the kind used in recruitment and the negotiation of employee workload and remuneration. PD’s often contain a listing of duties (a Statements of Duties), typically annotated to indicate the percentage of the position’s time assigned to the performance of each duty. It is a relatively straightforward task to assemble this kind of data, time consuming perhaps for groups that have not already done so, but well understood in the business community. Duty statements however, will probably not contained detailed accounts of what is transacted between whom. The PD therefore is merely a starting point from which the Analyst begins to identify roles and their transactional relationships. These may be easier to elicit when the analyst is armed with a list of candidate instruments, being those data-object-like things passed between roles in some way. The analyst at this stage can make use of questions based upon those listed in Table 4.3, but refocussed towards identifying possible instrument transactions.

AT has a (perhaps undeserved) reputation for being difficult to understand and use (Kaptelinin 1996, Fjeld 2002). It is entirely possible that an analyst could unsettle or confuse a stakeholder client by immediately adopting unfamiliar AT terminology. It is not reasonable to expect any client to understand how to follow Leon’tev’s implied ordering. It is crucial for user oriented analyses to establish credibility and secure user participation. An emphasis upon starting with readily available and commonly understood information phrased in common business terms, will not only make the Analysts’ task more straightforward, but should ease acceptance, engagement and participation among the stakeholder group.

Both organisational charts and a collected set of PD’s, though not traditionally described by AT, may be considered to exist approximately at the activity network
level. They portray components and dynamics of the group. Note that whilst positions often exist as documented institutional titles in some form, roles might not.

Individual PD’s however, have some currency between the activity and activity network levels. There is a sense in which positions indicate stations in the final design, as each station clusters interface(s) for a given ‘user’. A user sits at a **Station**. Defined in section 4.11.1, a station is a set of nested interfaces, needed for each of the activities conducted by a given user.

Stations represent the interface(s) by which one individual group member (a user) may perform as one or more subject(s) for one or more activities, according to the allocation of duties within the group. Recall, each activity, by definition, has a single subject, though that subject may perform multiple activities. There is no requirement for all the activities within a given station to share an identical subject however.

From an organisation’s structural point of view, the combination of duties instantiated within one individual has been defined as their Role (section 4.15.1). There is no reasonable expectation for all organisations to be able to specify exactly what a duty does or how it is performed (although accreditation to the ISO:9001 standard requires exactly such detail, in some form). A listing of member doings may well exhibit a part-whole relation to a listing of their duties, and thus may not easily map onto each other at all. In the final AT-flavoured design, stations present a combination of doings which may not closely resemble the group’s statements of duty.

![Diagram of Roles and Doings](image)

**Figure 4.30: Duties to Doings. Roles after Redesign**

To resolve confusions, it is proposed that terms be divided temporally, between a pre and post activity-centric interpretation. A stakeholder group will probably present
initially to the Analyst with a list of positions and of duty statements (roles). After analysis, this method’s BPM will be represented according to doings. Even before any refinements in the redesign phase, considerable mismatches may arise. Figure 4.30 illustrates how organisational roles may not even appear on an activity-centric interpretational plane. Roles may be considered analogous to sets of activities, but their clustering by arbitrary previously defined organisational duty may not closely resemble their clustering by doing.

As a starting point, when analysing the stakeholder group’s extant process, an initial identification of candidate roles may be deduced from their organisation chart and PD’s. These duty statements will assist the analyst to identify candidate instrument transactions and actions for further analysis, ahead of redesign. There is no realistic expectation for original organisational structures to remain invariant under activity-centric analysis and redesign.

4.22.1.1 Towards Disambiguation

It is not possible to construct an exhaustive list of all variations in structural philosophy and the rigour of application in organisations an analyst might encounter. Situations are likely to arise where mapping the stakeholder group into AT terms is a difficult exercise due to ambiguities of structure and vagaries in rigour.

Even should a stakeholder group rigorously employ commonly understood organisational structures and procedures, there will likely not be any uniquely correct mapping to AT terms. There will always be grounds to justify consultation with stakeholders in drawing up a consensually agreeable activity-centric BPM.

Should a stakeholder group apply an unexpected structure, and/or exhibit poor rigour, there are likely to be confusing many-to-many interrelationships in the hierarchy and omissions in any listing of duties. The analyst will need to be careful not to become enmeshed in such tangled scenarios. Hierarchic ambiguities (issues of abstraction and refinement) may greatly hinder any business process mapping. The analyst will need to keep some broad heuristic principles in mind in order to impose order upon the chaos.

As with all previously identified attempts at the construction of prescriptive AT SAD-like methods, hierarchic ambiguity and confusions within Leon’tev’s sliding scale have dogged the construction of this method. Even a prescriptive method designed specifically to avoid these pitfalls can hesitate in the face of a stakeholder group’s structural irregularities. To assist in the resolution of ambiguities, it is necessary to offer up some more concrete definitions, informed by the trial case.

First to consider are terms which to describe a stakeholder group as first encountered by an Analyst, when its processes and members are most probably described by title and some indistinct notions of ‘duty’:

**Duty:**
Components of positions, often somewhat arbitrarily assigned in the stakeholders’ organisation chart through position descriptions.
Stakeholder groups without documented statements of duty can usually produce information of this type under fairly straightforward elicitation. Variations in duties can differentiate otherwise equivalent seeming positions.

**Role:**
Ideally, a role would be a set of irreducibly linked activities; that is, it would represent the subject of one or more activities. However, as organisations are unlikely to have already adopted an activity oriented hierarchy, roles will simply be arbitrary place-holding titles for sets of duties, most likely elicited from position descriptions.

Roles can suggest probable locations for activities in well designed groups. Inefficient organisational arrangements and indistinct organisational taxonomic nomenclature however, mean that roles may not survive unchanged through an AT analysis and redesign.

**Position:**
In effect, an individual human group member. Positions may be described as a set of duty-oriented roles, or as a single ‘user’ of the System. Positions are extractable from the stakeholder group’s organisational structure. Position’s title and description would typically be drawn from the organisational chart for the stakeholder group.

The original duty-oriented terms may not retain their currency after being recast by Activity. To avoid ambiguity, it is recommended that new post-analysis terms be adopted:

**Doing:**
A collective noun for deliberate tasks undertaken by a person. The term ‘doing’ is deliberately intended to span across all AT layers (process, activity, action, operation) to allow discussion before analysis is complete.

**Subject:**
As defined in AT, a Subject is the protagonist (do-er) of an activity. Any activity may only have one subject, though multiple individuals may be authorised to play that part.

The pre-analysis duty-oriented term ‘role’ is often an initial indicator of activity sets with common subjects. After re-design, activities will be clustered by ‘user’ into stations.

**Screen-Set:**
The Interface(s) used by a Subject in the conduct of a single Activity.
Station:
The cluster of Screen-Set Interfaces by which one individual Group member (user) performs as the Subject (protagonist) for one or more Activities.

Clustering is allocated as doings coincide with Group Positions; this may not accord entirely with any pre-analysis statements of duty (Roles).

Despite these working definitions, consideration of the trial case gave rise to concerns in representing extant group structures according to activity orientation. It became necessary to consider the potential complications in addressing group structures.

To further reduce ambiguity, following the definitions offered above, the AT taxonomy was slightly modified to rename the protagonist at layer 2 as ‘actor’. The term ‘role’ was no longer suitable, carrying as it does a specific meaning related to a pre-AT-analysis depiction of the group and its process.

Table 4.8: Revised AT Taxonomy

<table>
<thead>
<tr>
<th>Doing</th>
<th>Facilitator</th>
<th>Driver</th>
<th>Product</th>
<th>Protagonist</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Activity Network</td>
<td>System</td>
<td>Agenda</td>
<td>Process</td>
</tr>
<tr>
<td>3</td>
<td>Activity</td>
<td>Tool (Set of Screens)</td>
<td>Motive</td>
<td>Outcome</td>
</tr>
<tr>
<td>2</td>
<td>Action</td>
<td>Screen</td>
<td>Goal</td>
<td>Transaction</td>
</tr>
<tr>
<td>1</td>
<td>Operation</td>
<td>Switch</td>
<td>Condition</td>
<td>Change</td>
</tr>
</tbody>
</table>

Table 4.8 indicates where the term for the action level protagonist was changed from Role to Actor, in recognition of potentially short-lived usefulness of the term ‘role’. As given above, roles are now defined as sets of irreducibly linked activities; that is, a contextual set of the subjects of one or more Activities, but they retain currency only should these duty-centric entities closely match the doing-centric stations.

4.22.1.2 The Relationship of Positions to Roles

Although the initial relational database deployment was abandoned (section 4.15), the sheer volume of information involved in eliciting even a small-scale real world business process suggested a database styled solution. To this end, during the trial case investigations, entity relation (ER) tables were assembled using the software product Microsoft Access. These greatly assisted in establishing what information might be readily available from the stakeholder group, in accordance with the reviewed workflow discussions above (Section 4.22.1).

As shown in Figure 4.31, a default one-to-many relationship was established between positions and roles, indicating that any one individual might be assigned several duties. This situation would arise when, for example, a university lecturer functions as a teacher, a researcher, a consultant, a committee member, an exam supervisor and so on.
As given above (Section 4.22.1.1), positions may be initially described as a set of duty-oriented roles. Some organisation structures may present many-to-many relationships between positions and roles. These may indicate opportunities for refinement. Such situations may indicate that imprecise duty statements obscure redundant or inefficient doings. The ability to identify, describe and even address such difficulties is a chief benefit expected of an activity oriented SAD method. Even if this kind of confusion is not apparent, multiple individuals may hold seemingly equivalent positions (eg: there are likely to be multiple lecturers within any one university faculty), without necessarily sharing all the same roles (Not all lecturers act as supervisors, course coordinators, committee members or exam markers).

In cases where multiple individuals hold seemingly identical positions; then, should their duties prove genuinely identical, the analyst needs to consider a representative example, but must make note of how many there need to be and their relationship(s).

![Figure 4.31: Some Entity Relationships from the Trial Case](image)

Should there be significant variation between the duties assigned to the individuals sharing a nominal position title, then the analyst should consider them as different positions (for the purposes of activity oriented analysis). A simple subscript type nomenclature may be adopted to indicate these. It may ultimately arise that the analyst suggests a group consider adjusting its organisational chart and or position titles to more accurately reflect the reality of it’s ‘doings’. Significant discrepancies of this sort, where the organisation is effectively ignorant its members’ doings, are likely to indicate process inefficiencies.

Should multiple instances of a position (and its roles) survive AT analysis and redesign, they are expected to yield multiple instances of a station. The analyst must carefully record all such repetitions. It is important to know if multiple simultaneous instances are permissible (according to the driving constraints of the group’s agenda), just as it is important to design for the segregation of access control by user. Such issues, though strictly matter of technical implementation, pass be recorded in
the specification to inform the design of security functions, the dimensioning of system resources (storage, memory, bandwidth, workstations) and so forth.

Each user will ultimately attend a station of the system. Stations represent the interface(s) by which one individual group member (a user) may perform as one or more subject(s) for one or more activities, according to the allocation of duties within the group. If there is a tolerable mapping of duties to doings, then positions may be expected to closely map to stations.

If the analyst and stakeholders decide to reorganise the group to enhance efficiency, then the originally identified positions may need redefining to better fit the final process design (and it’s facilitating system). In this case, stations will give rise to new positions.

During the initial analysis of the group’s process, identification of roles from the organisation chart and member duties is very important. Candidate instrument transactions will be initially identified as moving between roles. After the application of the simplifying heuristics and redesign, roles would typically become somewhat abstract sets of screen-sets (activities) commonly shared between otherwise non-identical stations (as indicated above in Figure 4.30). Ideally, a role should roughly equate with the station (or a subset thereof) but as organisation charts are typically not derived according to AT, this is unlikely.

The degree to which the initial roles do not accord with stations indicates the degree to which the organisation chart does not accurately reflect the process (at least in its post redesign form). Should a catastrophic misalignment between the organisation chart and the stations of the redesigned system interface impede client acceptance, then either the organisation’s position descriptions should be redefined or the system stations themselves must be split, merged or modified to align more closely to customer expectation. A client who prefers the latter will, most probably, retain many inefficiencies likely to have been identified by the AT analysis.

It is important to note that positions and roles are initially elicited largely because the information is available and understood by the client. Together with candidate Instruments, they serve as the basis for identifying action level transactions. Roles, initially useful as suggestive place holders for actors (and even subjects), are likely to lose currency, and so have no place in the AT taxonomy employed by this method. Difficulties encountered in sorting through a stakeholder group’s structure must not be allowed to distract or hinder activity-centric analysis. Indeed, such difficulties only highlight a need for a thorough analysis of the group’s doings, and may indicate the group is a strong candidate for restructuring.

There remains some scope for an analyst to offer varied mappings between the stakeholder group’s previous structural hierarchy and those used in the 5S method. AT provides a unique activity-centric perspective on the doings of human groups and the analyst may find the stakeholder holding to a position-centric, hierarchic or even user-centric conception. It may be necessary for the analyst to engage the group to ensure a consensually agreed BPM may be generated.
Chapter 4 – Method Building

It may prove necessary for the analyst to convince the stakeholder group to adopt an activity orientated perspective. An important benefit for the stakeholder group is the flexibility and resilience it brings to system design. Doings can remain largely invariant under changes in organisational membership, positions and roles. This notion will be explored in greater detail below. (Section 4.22.3)

Having relatively easily established both positions and thence roles (each of which is one or more subjects), from organisational charts and PD’s, the analyst should now start to identify instruments, with a view to identifying their transactions between roles.

4.22.1.3 Candidate Instruments

Following the path both of least resistance and of maximised stakeholder participation, the analyst must continue to prioritise that information which is already available. The trial case highlighted that even a poorly organised and entirely manual Process will still produce, use, consume, store, recall and delete things rather like Instruments.

Almost by definition, any group process will already transact data-object-like things in one form or another. After establishing roles, the analyst will need to decompose these duties into transactional doings whereby an actor (probably closely associated with a role) produces, consumes, stores, retrieves or modifies these data-object-like things. These may be as mundane as time-checks and simple schedules or as complex as multiple-order invoices and balance-sheet analyses. Some of these may be entirely intangible (such as audible time signals, tacit knowledge of process, delivery of training and personal conversations etc), some may exist in paper or ‘token’ form, and some may exist as data in existing computer systems (such as spreadsheets, sales records, inventories or document templates).

The analyst should identify as many candidate instruments as possible from existing files, folders, forms, records, registers, lists as possible. Many of these may ultimately be adopted and modified into genuine instruments, where one of the parties in the transaction will be the computer system under design (data-items, flags, variables etc). It is probable that user acceptance of a new system may increase if, so far as is feasible, instruments bear resemblance to preexisting data objects.

‘Virtualised’ (electronic) interpretations of previous paper documents should probably exhibit visual similarity. This would be in accord with Neilsen’s heuristics for user interface design, which recommend a “match between system and the real world” (Neilsen and Molich 1990) and Grudin’s third consistency definition; which is “correspondence of interface features to familiar features of the world beyond computing.” (Grudin 1989).

Each of these transacted data-object-like things represents a potential Instrument. Whilst many of the identified candidate instruments should quickly become apparent from the group’s current suite of standard documents and procedures, some may need to be inferred or actively elicited as they may be perceived as peripheral or even tacit by group members.
It is important to identify the authors, producers, handlers and consumers of each candidate instrument. It is also important to identify and record Transaction Constraints, particularly temporal (issues of sequence) and deontic (issues of obligation). The analyst must identify instrument transactions between roles as fully as possible, as these identify actions. It is anticipated that, as many instrument transactions may be poorly understood within a group and that many transactions may have become redundant or inefficient over time, some iterative consultation may be required to reach a consensually agreed picture of current process. This is a common issue for business process mapping (Robertson & Robertson 1999, Chung et al. 2000).

Identification of instrument transactions however serves chiefly to assist in the identification of actions. This requires sufficient transactional data to identify Actors and subjects, to allow conflation into activities. Ultimately, the system and its interface(s) will be specified on a per-user (station) basis. This requires a mapping of the organisational structure into an activity network, which is positions and roles must be recorded in initial elicitation and analysis. The analyst must take care not to become distracted by complexities in these details arising from unusual or poorly described organisational structures. Ultimately, only sufficient detail is required to permit feasible re-assembly (clustering) of the interfaces into per-user stations in a manner acceptable to the stakeholder group.

It becomes necessary to refine the conception of an instrument transaction.

### 4.22.1.4 Three ‘T’ Terms Become One

At a previous stage in the development of the method (section 4.16.1) the narrative nature of information which could be elicited from stakeholder groups informed the notion of Story Fragment Analysis (SFA). During the brief period in which the SFA notion was entertained, three principle functions were identified; transaction, transformation and translation (Figure 4.18). In this notion, transactions existed between activities and the system; transformation loosely described the method’s decomposition analysis and translation occurred when operations were somehow mapped to interface widgets.

Although the SFA notion has long since been abandoned, the three ‘T’ terms persistently suggested themselves for describing instrument exchanges in more detail. Indeed, for a very brief time the list grew to include such unlikely terms as transmute, transmit, transport and transfer, as the method’s ancillary working taxonomy struggled to keep pace with changing perceptions of group dynamics.

It quickly transpired that only two fundamental things could happen to Instruments.

Firstly, Instruments could be moved between actors. As indicated by the pipe and joint reduction heuristics employed in the trial case (section 4.21.2) in the case where an actor moves a candidate instrument from itself to itself, reduction is strongly indicated. The storage and recall of an instrument, together with any changes in its security status (such as it’s access control) are aspects of it’s movement, described by temporal and deontic constraint conditions, and reflected in the driving Goals of the actions which handle it.
Secondly, some form of work may be performed upon instruments, such as creation, deletion, editing, joining or splitting. As also indicated by the pipe and joint reduction heuristics employed in the trial case (section 4.21.2), should an instrument pass through an actor’s hands unchanged, then reduction is strongly indicated.

Only two terms were therefore of use, transaction (*movement*) and transformation (*work*), and all others could be reduced to these.

The term transaction however, may be deemed to contain the notion of transformation as a subset. Any instrument transaction would contain some indication of movement between active parties, and also an indication of work done upon it. The three (and briefly, six) ‘T’ terms were thus promptly reduced to one.

It is extremely important to note however, that the analyst must be most careful to elicit and record the transformational aspects of all transactions, including driving goals and any temporal or deontic constraints.

In eliciting and recording an instrument transaction, the analyst must ensure that following details are captured: the actor(s) it comes from (under conditions, circumstances, constraints and obligations); the actor(s) it goes to (under conditions, circumstances, constraints and obligations); the transformation it undergoes (create, delete, edit, join, split etc) and the driving goal (purpose) sought in the conduct of the transaction.

These may be used to revisit the suggested list of elicitation questions used to guide the analyst. Table 4.9 presents a new suite of questions which would accord with these considerations.

There are two primary functions for the analyst to perform with this data; construction of the extant activity network (BPM) with its instrument transactions, and collection of the goals and constraints for each transaction. To this end, question 1 (positions) is recorded as the list of individuals (strong candidates to be the users for system Stations). From these, question 2 elicits duty-based roles (initial candidates for actors). Questions 3 and 4 assist in identifying candidate instruments, and their transactions between roles. Questions 3a and 6a specifically tie the roles and candidate instruments together into the earliest forms of the directed graph activity network.

Questions 3b, 6b and 4 are particularly important. These describe the true nature of each transaction and record the current doings of the extant process. This information should be subsumed into question 5 (Goal).

The general form of a goal statement should be: *take Instrument(X) from Role(A), perform transform(T) upon it under conditions(C), and pass the resulting Instrument(X’) to Role(B)*. Complex goals with multiple instruments and multiple supplier and recipient roles are possible of course. The constraints (temporal and deontic conditions) should indicate timing, sequence, ordinality, cardinality and obligation.
## Chapter 4 – Method Building

<table>
<thead>
<tr>
<th></th>
<th>Question</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Who are the members of this Group?</td>
<td>(Positions, individual)</td>
</tr>
<tr>
<td>2</td>
<td>What duties does each Position perform, what is their Position description?</td>
<td>(Roles, duty)</td>
</tr>
<tr>
<td>3</td>
<td>What information does each Role require to perform their duty?</td>
<td>(Candidate Instrument(s) in)</td>
</tr>
<tr>
<td>3a</td>
<td>From whom (Role) do they obtain this required information?</td>
<td>(Role(s) from)</td>
</tr>
<tr>
<td>3b</td>
<td>Under what conditions (timing/sequence, circumstances, obligation)</td>
<td>(temporal, deontic constraints)</td>
</tr>
<tr>
<td>4</td>
<td>What do they do with this information?</td>
<td>(transform)</td>
</tr>
<tr>
<td>5</td>
<td>What is the purpose of this transaction?</td>
<td>(Goal)</td>
</tr>
<tr>
<td>6</td>
<td>What information does each Role return, send on, or otherwise dispatch?</td>
<td>(Candidate Instruments out)</td>
</tr>
<tr>
<td>6a</td>
<td>To whom (Role) do they send this information?</td>
<td>(Role(s) to)</td>
</tr>
<tr>
<td>6b</td>
<td>Under what conditions (timing/sequence, circumstances, obligation)</td>
<td>(temporal &amp; deontic constraints)</td>
</tr>
</tbody>
</table>

### Table 4.9: Questions to Assist in Eliciting Transactions

Goals will detail exactly what is to be done to, with or upon each Instrument as well as when and for whom. Though it is beyond the scope of the current project, Goal statements should ideally employ some form of multi-modal logical operators to facilitate constraint-satisfaction analysis. At the least, Goals must express a doing in some structured linguistic terms, perhaps similar to pseudocode forms. These structured phrases inform the ‘functions’ (transforms) of the final computer System. The greater the precision and rigour with which Goals are recorded; the more reliable the specification passed to coders for construction and implementation of the design.

It is worth noting that although the term ‘transaction’ appears in client-server architecture literature (Maciaszek and Liong 2005), its use in this research project should not present a significant problem. Indeed, there may be some unintended synergy between the two uses, as all instrument transactions retained under the 5S method after redesign are actually transactions between users (analogous to clients) and the system (analogous to server). Exchanges of instrument-like objects purely between humans occur outside the scope of the final system specification. Details of such exchanges are retained however, as they provide contextual settings which inform design decisions. Any instrument transactions purely between the system & itself are doings which have been ‘automated’ and, no longer classifiable as 5S transactions are actually internal data-object transformations.
4.22.1.5 Navigating Leon’tev’s Labyrinth

The disambiguations offered above, dividing terms between pre and post AT analysis, should go some way toward addressing ambiguities arising from stakeholder organisational structures. Considerable difficulties remain within the nature of AT itself.

Reflecting upon the method’s construction, detailed throughout this chapter, and upon those few preceding research attempts to apply AT in a similar manner, it becomes apparent that Leon’tev’s AT levels (Leon’tev 1978, 1981) provide at once the most powerful and least practical aspect for building prescriptive AT methods. The sliding hierarchy is highly dynamic and contextual, providing capacity to reflect the vagaries of humans’ engagement in complex activities, but that adaptability makes it difficult to apply.

The trial case highlighted difficulties in interpreting business activities in AT terms. In particular, it is rarely clear if an observed doing is an activity or an action. As observed earlier (section 4.14.2), Vrazalic (2004) applied the term ‘activity’ in a sometimes inconsistent manner, sometimes it seems to relate to the entire process (‘Student Obtains a Degree’) and sometimes to a subset (‘Take a Subject’). The nature of AT itself is largely to blame for such irregularities, as it invites such flexible application. Perhaps in response to the difficulties, Vrazalic too quickly departs from AT and finds an ancillary theoretical base in Distributed Usability.

To apply AT consistently across a group’s process and avoid its inbuilt pitfalls, it is necessary to ‘pin it down’ in some way. Though, as discussed in section 2.6.7, Bardram (1989) offered some guidance in telling conditions from goals, there has been little sign of the notion being operationalised. Initially, this project applied a standard taxonomy and the use of an overall activity network level of abstraction. Two further mechanisms for ameliorating this classic AT hierarchic ambiguity effect now suggest themselves: Firstly, to consider a temporal snap shot of the group’s process, wherein users are locked into their sets of roles. Secondly, it would seem appropriate to impose a new procedural rigour upon AT itself (when deployed as the informing theoretical base for a SAD method). A more prescriptive means of identifying and classifying actions and activities (in particular) is required.

The taxonomy and extended hierarchy helps prevent ambiguities at the highest (process) and lowest (operation) levels. A prescriptive approach will assist to resolve difficulties at the action and activity levels.

The trial case highlighted a need to elicit initially from the non-AT based information available from the stakeholder; positions, roles and candidate instruments. From these the analyst identifies transactional relationships between the roles, as instruments are created, exchanged, modified and consumed.

The activity network has been described throughout this project as a directed graph. There has been some vacillation between calling the nodes activities or actions. The reality is that they must ultimately be activities, but that these are assembled from actions, which are more easily identified. This suggests a workflow which may not directly follow a linear path along down the sequence implicit in Leon’tev’s hierarchy (as suggest in Section 4.22.1).
Transactions of candidate instruments between roles may be represented in a directed 
graph. It is here proposed that these should be identified as goal driven actions. As 
the analyst applies refining heuristics, some of these actions may be modified. It may 
be that some will be conflated, suggesting they were actually of a lower order, and 
others may be split, suggesting they had been of a higher order. The analyst however, 
should seek to identify action layer transactional events.

The analyst first assembles a rough outline of an activity network (the most abstract 
of the four AT layers). Following the path of least resistance and making the best use 
of available information, the next step however, is to temporarily bypass the activity 
layer, seeking actions. The directed graph of initially elicited candidate instrument 
transactions could be non-trivial, with nodes of high order, receiving and sending 
multiple Instruments with multiple other nodes. The analyst should seek to clarify the 
network into a set of action nodes with instrument transaction arcs. This will present 
a functional BPM of the extant process. Each action node must be accompanied by 
its goal data. As described above (section 4.22.1.4) these goals statements must list 
the Actions from which instruments are received and those to whom they are sent, 
together with all temporal and deontic constraints.

It is now necessary to split each of these actions into Single Instrument Nodes 
(SINs). As indicated in Figure 4.32, an action is divided into as many SINs as it has 
Instruments (in and out). Each SIN must carry a subset of its parent action’s goal 
statements. The goal statements of each sibling SIN should indicate (almost 
genetically) the familial and causal links between them.

The analyst is now in a position to conflate these SINs into activities, which 
constitutes a shift back up the AT layers, towards abstraction. The rules for 
identifying activities are extractable from the precepts of AT itself. By definition, an 
activity has both a single subject (protagonist) and a motive (driving purpose).

Formed correctly, each SIN has an actor protagonist and a goal statement. There is 
also a suggestion of possible activities in the earliest elicitation of positions and 
roles. So far as is feasible, the analyst may strive to enhance user acceptance and 
system learnability by striving to follow these original groupings. It may prove 
necessary, however, to radically depart from the extant model, partially as a 
computer System is envisaged to automate and facilitate, but also as inefficiencies 
and redundancies are uncovered and addressed.

Aside from the suggestion of activities contained in the pre-system BPM, two 
important heuristics can guide the analyst to identify activities. Firstly, as any 
activity has only a single subject, SINs may be sorted by their actors (role). At this 
stage it proved the case that some duty-oriented roles did not match well with the 
reality of their doings.

In a second sorting, the analyst must examine the driving goal statements for each 
SIN. AT specifies that an activity has a motive (driving purpose). The motive of the 
activity may be seen as a conflation of logically consistent action-level goals. Goals 
which cannot be consistently conflated (most probably due at times to 
inconsistencies in their temporal and deontic constraints) will not be members of the 
same activity. In the trial case (section 4.21.2), it was seen that student receives
assignment could not be collapsed (conflated) with student submits assignment because of the conditions and constraints associated with those actions.

Figure 4.32: Decomposing SINs

Figure 4.33 illustrates the conflation of SINs into activities. Resultant activities must each have a single subject (who may also be subject for other activities) and a consistent motive.

(The conflated motive may be a complex string of structured text. It may prove necessary to preserve the goal statements of each component SIN separately, probably in some database (or similar) format.)

In this earliest expression of the 5S method, there is a need for some interpretational skill on the part of the analyst, and iterative stakeholder consultation may be required. In future research however, techniques drawn from the fields of discrete mathematics and multimodal logics may lead to some degree of automation. If goals and other constraints may be recorded in some appropriate language, then multimodal consistency checking and constraint satisfaction techniques may greatly reduce the need for the analyst to test-fit their findings against stakeholder feedback. It is thus vital that the constraints and conditions ‘inherited’ in each SIN must be accurately recorded in a structurally consistent manner.

Running these two sorting heuristics (with stakeholder involvement where appropriate) subjects may, or may not, retain their original structuring given from the
duty statements in the organisation chart. Positions, roles and even goals may be radically re-engineered as it is considered how a computer system can improve efficiency in the business process.

Figure 4.33: Conflating Activities

The structured motives (and their component goals, with conditions) will provide sufficient information to identify candidate operations (whose conditions are derivable directly from the temporal and deontic constraints). These should inform the selection of appropriate interface components (switches). The switches are clustered into screens, screen-sets and stations according to the conflation of activities. Each individual (position), perhaps with a newly redefined role, will operate a station.

4.22.2 The Importance of Retaining Activity Orientation

It is the central tenant of AT that the activity is the fundamental unit of analysis. Throughout the construction of this method it has been necessary to return to this principle several times. This guiding principle has led the recovery from each pitfall, false start and hesitation.

As observed above, stakeholders are unlikely to have an activity oriented process map of their groups’ process, nor any idea of how to prepare one. The reviewed workflow concentrates initially upon eliciting information that is readily available from the stakeholder group. This may, however, produce hierarchic confusions between positions, roles and subjects.
The important principle to be followed by the analyst is that the activity must always remain the fundamental unit of analysis. The activity, however, is not simple to read from the existing group’s process. In identifying positions, roles and the candidate instruments they transact, the Analyst must not become lost in hierarchic quandaries. These details allow the analyst to identify actions and then assemble the activities prior to redesign.

Subjects are activity level protagonists, and roles are sets of subjects, at least in as much as they coincide with initial stakeholder positions. Ideally, roles should closely align with stations, though either stations or (more properly) roles may require refinement to better fit group doings (depending upon stakeholder flexibility). It is reasonable to expect that overall agendas (the purpose and functions of the group) are largely invariant over time. If a stakeholder group changes its membership, rearranging positions, then it is likely that what gets done (activities and their subjects) are less susceptible to change than are their arbitrary statements of duty (organisational roles). A post-redesign reconfiguration of a group’s structure should require little more than a re-clustering of subjects and their container stations; the majority of the SAD work should retain currency.

While it remains vital to identify a group’s individuals and their hierarchic organisation. Such data informs crucial system design constraints such as interface clustering, transactional dimensioning and secure access control, the 5S analyst-designer, however, should recognise that initially, their work concentrates on roles more than on individuals. Positional data, elicited in the earliest stages and subordinated to the identification of actions, must be retained to inform the redesign and recomposition of the system.

Before the analyst can be sure of their BPM, it is necessary to consider aspects of instrument behaviour indicated in the trial case.

### 4.22.3 Refining Terms

Instruments are crucial objects, as their transactions are the workings of both the BPM and of the system designed to enhance the process. Considerations arising from the trial case highlighted some issues regarding instruments, including possible variant forms.

#### 4.22.3.1 Of Chickens and Eggs

The curiously contra-intuitive pathway adopted by this method through Leon’tev’s hierarchy, opens scope for confusions of cause and effect, whereby property inheritance between AT levels becomes unclear. It is important to note that instrument transactions, are used to identify actions and SIN’s, and their constraints are used to conflate activities. Instruments are thus entities that exist across levels, which largely define actions and SINs.

Once conflated and identified however, activities obtain a special status. They are, by definition, the fundamental unit of AT analysis and take on a semi-detached, almost stand-alone quality. Once defined, the instrument transactions associated with an activity may be greatly modified without necessarily changing or redefining the
activity itself (except in so far as its motive statement must reflect its doings). The activity network may then, at last, truly resemble its AT conception.

Under redesign, the 5S method strives to redirect transactions so that they pass across the system boundary. Those candidate instruments which do so become true instruments, and appear as data objects in the System Data Repository (SDR), as in the trial case. The ‘bending’ or redirecting of transactions need not necessarily change the identity of the activity nodes themselves (though they may change if required).

Any changes to an activity, including any redirection of its transactions or indeed, activity merging, splitting, creation or deletion, will necessarily entail changes in its motive statement(s). Changes to an activity will, of course, implicitly change component actions though as the activity is the fundamental unit of analysis, there is no need to fully elucidate such changes. Even should an activity change involve a radical rearrangement of sub-doings, all component actions (and their goals) will by definition retain their commonality of protagonist (actor/subject) and consistency of driver (goal) with the larger motive.

The analyst/designer should appreciate that at this stage, actions can be deemed somewhat fluid. Analogous to a metamorphosing butterfly, all the internal organ ‘ingredients’ of the larvae enter the enclosing cocoon, but their final form may be quite different. It is quite secondary how the new creature’s organs function, so long as the result is a butterfly; a need to investigate them simply requires investigation to see how they have been rewired and rearranged from their original form. In a similar way, Actions can be ‘read off’ from the finished activity as required, being merely subsets. In such a case, it is recommended that actions be extracted in their primitive SIN component state, as these comparatively simple elements may be extracted with minimal confusion.

In the final design, the chief purpose of understanding lower level entities such as action (SIN) and operations is to identify the clustering of sub-doings and their Interfaces into usable stations, designed to suit the stakeholder group’s organisational structure so far as would be feasible for the redesigned process and acceptable to the client users.

4.22.3.2 In and Out

A chief aim of the 5S method is to produce a set of specifications which may be passed on to coders and interface designers for the construction and implementation of a facilitating computer system. To best enable this, it is desirable that the specification be as standardised as possible, with the least potential for confusion or ambiguity.

Some remaining confusions surround Instrument transactions. Any number of instruments may enter or leave an activity. Any single Instrument may be described either by its sender or its recipient, and any single activity could be discussed in terms of its obligation to send, or its need to receive. This invites causal ambiguity.

To simplify and clarify specification, it was decided to consider chiefly the ‘need to receive’ an instrument. If each activity’s motive contained a consistent set of
transformational goals, then instruments may be deemed prerequisite ‘ingredients’ for satisfying this motive. The obligation of an activity to supply output to another Activity may be ‘read’ from the recipient activity’s ‘need to receive’. Each activity may be considered to seek its required elements, in order to create one or more output objects, which are then simply available (under conditions) to other activities as required. Issues of timing and obligation (temporal and deontic constraints) are retained, but rephrased as necessary.

In redesigning and refining the process, the analyst/designer should strive to re-express all ‘send to’ transactions as ‘get from’ transactions. The activity network may then be visualised as a series of activity nodes each acting as resource pools, each with trailing ‘get from’ connectors to preceding activities. A ‘get from’ requirement neatly describes any need to consult an external resource, system or tool. It should be noted that some ‘send to’ transactions may remain, particularly if an external agent/actor requires a system output, such as submission of a bid price or return of a value to satisfy a query.

It is an interesting vector for future research that this notion of expressing one’s outputs and another’s inputs, may suggest that the expression of many deontic (obligational) constraints may be converted or subsumed into purely temporal (sequential) constraints. Such an idea may yet constitute a considerably powerful simplifying principle.

4.22.3.3 Meta, Sub, Template and Transitory Instruments
Hierarchic ambiguities seemingly plague attempts to impose prescriptive method. The data-object-like things (instruments) transacted between actions, SI/S and activities are not immune to the effect.

Consider a doing such as “write an exam paper” or “update the curriculum”. An exam consists (normally) of more than a single assessment task or question. Should the writing of each question be a separate doing? Is the exam itself an instrument, or each question? A curriculum typically contains descriptors for a number of academic subjects, topics and learning outcomes.

Whilst the questions with the exam might be considered mere components of a single Instrument (analogous to the characters and words within this sentence), the academic subjects within the curriculum certainly have individual status, being conducted as standalone modules. Should the 5S method therefore encompass the notion of container instruments (meta-instruments), or perhaps borrow a notion from Object Orientation (O-O) and adopt a notion of instrument classes, whereby they can be grouped according to characteristics, function and or taxonomy?

Consider an entity such as a corporate website. Is that a single instrument? What of all its component text files, images, HTML files, CSS files and internal hyperlink ‘switches’? Should the 5S method consider all such things as component instruments (sub-Instruments), requiring explication only under certain conditions?

Although a great many candidate instruments may be identified in the BPM, and many more may be proposed during analysis and design (as illustrated in the trial case), only those ultimately transacted across the system boundary will be true
Instruments (appearing as data objects within the system, and accessed by users via the interface(s)). What, then, would it mean to have Meta-Instruments or Sub-Instruments?

A superficially enticing notion could be that meta or sub instruments may exist on different AT levels (perhaps meta instruments at the activity level, and sub Instrument at the operation level). This notion must be rejected however, before it reintroduces hierarchic ambiguities. It is the identification and analysis of the mechanism of instrument transaction, *outside* of the AT hierarchy in some sense, which allows the analyst to contain hierarchic ambiguities and conflate activities.

The analyst should strive to avoid the tempting notion of introducing meta or sub instruments. A great variety of transforms are allowable, whereby Instruments may be consumed, destroyed, modified, created, or merely consulted. Instruments are best considered to cohabit a common plane, joined through a great variety of transformational relations. That one instrument may inform the construction of another, or have a nature that depends upon the natures of several others is entirely allowable.

Consider that if a doing such as “update curriculum” were poorly described, an inexperienced coder (or some speculative automatic coding algorithm) may create excessively duplicated code for each instance rather than a more efficient modular function. The specification offered to the coder by 5S must convey the relations in a neutral manner, so as to allow any coding architecture to apply. Should a coder choose to adopt an O-O technique for example, the 5S specification should permit the clustering of instruments into object classes, but without *imposing* that approach upon coders who may use other paradigms.

A particularly tempting instance of a potential meta-instrument would be the template instrument. A blank record form for example which is used to format individual records (such as invoices or membership cards) may exist as a template instrument, whilst each *instance* exists as an instrument in their own right. Again, the analyst/designer is encouraged to avoid such a decision. The so-called template is indeed an instrument, but merely a resource (as with any other) to inform the formatting of another.

Consider that in a doing such as “confirm an appointment” it is probably necessary to consult some kind of calendar instrument. The calendar itself however, might not be changed. Any instrument may be required by any number of nodes, in any number of ways. It may be changed or not. In section 4.21.2 above, heuristics for graph simplification targeted instrument transactions which passed through a node unchanged. Such an instance does not occur if the instrument never leaves the node, even if it is unchanged. Regardless, those heuristics have served their purpose once the BPM is constructed. During redesign, once activities are conflated, instruments (as discussed above (Section 4.22.3.2)) may best be considered simply the required ingredients of doings.

An entirely separate activity would have created, or could modify the template itself, and constraints should convey any obligation for records to inherit the changed format. Alternatively, the format might only be applicable to records of a certain date.
range, or only when they are opened and displayed or printed. Such decisions fall to the coder, but are informed by well written constraints in the 5S specification.

Under the ‘get from’ oriented paradigm (section 4.22.3.2) activities which modify a template will have no obligation to ‘send’ advice to record-holding activities which maintain records. Under the 5S specification it is assumed that the record-handling activity will need to consult the template as necessary. Should the coder decide this involves inefficiently frequent polling between processes, then they may choose to reverse the obligation and code it as a one-to-many broadcast. This is an implementation decision, available after the 5S specification is delivered. Whilst 5S imposes a consistent internal approach to analysis, it must not oblige the coder to reject or adopt any specific implementation paradigm.

Another potential variant form of Instrument may arise when the doing, and its data, are extremely short lived. Consider a doing such as ‘consult an online map to determine travel distance’. The online map resource (such as GoogleMaps™ [http://maps.google.com]) would constitute an external tool but the value it returns to the query would be a short-lived instrument, consumed as soon as it is incorporated into some larger doing such as, ‘calculate shipping cost’ or ‘calculate employee travel loading’. (If the computational/financial/time costs of acquiring such an externally sourced instrument are sufficiently high, it may be advantageous to maintain a local equivalent instrument instead.)

The analyst/designer must resist any temptation to consider the value ‘special’ and to posit some unique form of transitory instrument. Yet again, the analyst/designer must simply faithfully record the ‘get from’ obligations and the goals and generate high-level specification for the coder. During construction and implementation the coder will decide how best to handle such short-lived data, according to whichever software paradigm and hardware is being used.

4.22.3.4 Instrument-Doing-Constraint Relations
Candidate instruments may prove to be better expressed as doings than things (for example, what may seem to be a meta instrument that stores other instruments may be better represented as the act of storage), or as constraints. Likewise, initially identified doings may be expressible as instruments or constraints; and initially identified constraints may be expressible as instruments or doings.

The analyst must maintain an open and flexible approach, especially in the earliest phases of the work, to find the simplest and most directly comprehensible representation of the group’s process and agenda.

4.22.4 Refactoring and Re-Engineering for Re-Use
As described in section 4.22.1.1, it is useful for the analyst to ‘pin down’ the group and its doings in time for the purposes of analysis. It must be recognised however that these things are, in reality, dynamic. Businesses change personnel quite often, and may modify their positions and duty-based roles at that, or any, time.

If the 5S method yields a specification tied to a frozen snapshot, it is important to ask how it would cope in a more fluid environment. Such consideration give rise to the
notion of specification re-use, refactoring and re-engineering. These are important concerns in Software Engineering (Sommerville 2001).

In considering the trial case, it became apparent that the 5S method exhibits considerable capacity to accommodate variations in the instantiation of roles (stations) in positions, particularly during organisational reviews and changes.

Once the system instruments have been defined, together with their transformations and conditions, switches may be chosen for each discrete operation. These switches are clustered together into action level screens, and sets of these screens assembled for each identified activity. Re-clustering these screen sets to accommodate different position configurations should therefore be a relatively straightforward process.

As a possible future refinement, to the 5S method it is conceivable that some high-level user of the client organisation might be granted ‘conceptual-level’ administrative access into the system structure, to directly re-cluster and reconfigure switches, screens and screen-sets into new stations, cutting, copying or pasting them into new positions with perhaps a graphical drag-and-drop interface. Considerable security checks and restrictions would be required to prevent accidental or malicious change, and enforcing internal organisational security protocols.

4.23 An Overview

There have been many workflow conceptions throughout the construction of the 5S method. This is as might be expected, as the method itself is no more than a prescriptive AT-driven workflow.

The consequences of the trial case, discussed in preceding sections, have highlighted the phased nature of the workflow. Specific terms, notions and categories that have currency prior to the AT analysis or the redesign, may not necessarily hold after either (or both) of those stages. A summarising overview of these terms and notions as they can change serves to inform an overview of the final workflow itself.

To more fully explain the changes in terminology, it is useful to consider, once again, the seven nodes of the Engström matrix as they might appear in the first phase - prior to AT analysis. In this phase, activities are probably not even imagined by the stakeholders and, at best, will be poorly described. Roles, if described at all, will likely be arbitrary statements of duty which contain limited (if any) details of actual doings. Positions should be extractable from some kind of organisational chart, or at least a listing of group members. Instruments will not be defined, but a preliminary candidate listing could be assembled from the documents, records, flags, permissions and other data-like things passed between positions during the conduct of work. Some elicitation is probably necessary to extract these. The group’s process will probably not be facilitated by any central computer system (though it may). It may be assumed that whatever cluster of facilitating tools exist are either not satisfactory, or are not well suited to what the group actually does; or else there would be little need to run the 5S method.

In the first phase, the seven Engström nodes may be summarised as follows:
Chapter 4 – Method Building

SUBJECT:
Initially identified from stakeholder’s Organisation Chart Positions.

TOOL:
All facilitating tools and techniques, physical or computerised or not.

OBJECT:
Motivating aim, best expressed as outcomes and outputs.

RULES:
Obligation and Timing constraints.

COMMUNITY:
Other Positions (Roles) involved in transactions.

DIVISION-of-LABOUR:
List of data-like objects required from other Roles. These are expected to largely be described using Deontic (obligational) constraints. These trace and describe the obligations to provide Instruments for specific transactions to occur.

OUTCOME:
Products of doings (intended or otherwise).

In the first phase, the Engström nodes carry values quite close to those defined in classical AT, only subtly skewed to apply to the specific case of a business model with transacted data-like objects passing between subjects. Recall that 5S adopts a tool centrism, as it seeks to define the specification of some central facilitating system. The specific meanings assigned to DivLAB and rules have been consistently applied in this since quite early in the construction of the 5S method, and could be considered not inconsistent with classical AT.

After applying an AT analysis, as described under the 5S method, the group’s BPM will have been described by its doings. This involves some clarification of the terms, and may be considered a second phase of their transformation.

In this second phase, an activity consists of sets of SINs conflated according to commonality of subject and the logical consistency of goals. Roles, probably still closely aligned with their position descriptions until late in the analysis, should be redefined in terms of their doings. The initial set of roles elicited from the organisation chart may lose much of their relevance at this point. Positions, however, remain as individual users listed in client’s organisation chart. At this time the set of individuals is assumed to be essentially invariant (the case has been frozen in time for analysis). The analyst may well see opportunity for reorganisation or redefinition of positions, but it is suggested that by preference, any changes take the form of role clarifications. Candidate instruments are still data-like objects exchanged between the conflated activities (one per SIN). The set of Instruments will have changed. Numerous may have been discarded as peripheral, numerous others may have been proposed as doing transactions are described and refined. The outline of a new facilitating computer system is described by the space between those activities whose
Instrument transactions could run across its boundaries (allowing it to facilitate transactional connections previously running directly between users).

In the second phase, the seven Engström nodes may be summarised as follows:

**SUBJECT:**
The protagonist, one per Activity.

**TOOL:**
SET of all tools used in Actions.

**OBJECT:**
Goal statements incorporating Instrument transforms, most usefully expressed in some structured language, possibly some paradigm-neutral pseudocode.

**RULES:**
SET of Conditions & Constraints applied to Instrument transaction. These are expected to largely take the form of Temporal constraints, indicating the sequencing and ordering of transactions.

**COMMUNITY:**
SET of Roles with which Instruments are transacted, indicating those from whom required Instruments are obtained, and those for whom output Instruments are required.

**DIVISION-of-LABOUR:**
SET of Instruments required for Transactions.

**OUTCOME:**
SET of Instruments available as output.

The activity oriented model is now expressed in AT terms, under a consistent taxonomy. Here the analyst has conducted a feasible and highly utilitarian business process mapping (a worthwhile outcome in its own right). Such a map is likely to meet many of the requirements for a business to achieve ISO:9001 Quality management compliance. The stakeholders and analyst should now be able to view the entire BPM and see how a facilitating system could be added to their tools (replacing or subsuming many of them) greatly enhancing the efficiency of their doings. This opens the way for redesign of the entire process, incorporating just such a system.

In this third and final phase, an activity assumes its proper place as the fundamental unit of AT analysis. Activities of interest are those which reside tangential to the system boundary, which exchange instruments with the system, and who’s tools incorporate elements of the system interface. The notion of the role is largely replaced by sets of subjects, conflated into stations according to the allocation of users within the group. It is quite possible that the original duty-oriented roles no longer match to the redesigned clustering of doings. Positions however remain, as indicators of group members, even if their roles are redefined. They are now users of
the system, Subjects for one or more activities, operating through custom designed interface stations. Instruments are now closely identifiable with computer data variables, stored in the system’s data repository, and exchanged with users through their station interfaces. The final form of instruments will depend upon the code-paradigm adopted under implementation; 5S will present its specifications in terms of instrument, to best suit the broadest range of technical options. The system itself is now visible as a computerised entity whose doings are well specified to a grey box standard. The DivLAB and outcome clearly list all instruments (data objects) that pass in and out. The object statements specify all transforms and processes the system must perform upon them; as well as their temporal and deontic constraints. The tools now indicate the clustering of interfaces according to action, activity and user, and greatly facilitate the design of an interface with high usability and (hopefully) high user acceptance.

In this third phase, the seven Engström nodes may be summarised as follows:

**SUBJECT:**
The user at their station

**TOOL:**
The hierarchic sets of switches, screens, screen-sets & stations

**OBJECT:**
Motive statement that encapsulates the set of goals. Contains formalised pseudocode-like statements of temporal and deontic constraints to inform coding.

**RULES:**
Should mostly have been converted into DivLAB entries, specifying all the required instruments for each transform.

For example: A temporal rule such as "process must be run on a Tuesday" becomes a deontic requirement that some instrument (data variable) nominally called TUESDAY has the value "TRUE" before it may commence. NB: The degree to which rules are converted into DivLAB indicates expected reliability of the final system.

**COMMUNITY:**
The system is now the entity with whom all instruments are transacted.

**DIVISION-of-LABOUR:**
Activity inputs, expressed as transformational requirements.

**OUTCOME:**
Activity outputs.
Table 4.10 lists all these common terms. Each is assigned three sets of values. The first are working definitions which apply to the stakeholders’ group and process as the analyst first encounters and describes it. The second indicates how the meaning of the term may change after the AT-informed analysis. The third set of meanings indicates terms after the redesign of the process, to accommodate the central facilitating computer system.

The change in meaning constitutes both a shift from the initial duty-orientation to the AT doing-orientation and also the changes made under redesign heuristics to specify the characteristics of a facilitating system.

Figure 4.34 presents an abbreviated graphical representation of the redesign process. Recorded against each of the seven Engström nodes are their second phase meanings. Highlighted outside each node is the effective meaning of the node after redesign. The conversion of temporal rules into deontic DivLAB is indicated, and the final assemblage of a system specification is indicated by the dotted lines.
Table 4.10: Method Terms Before and after AT Analysis and Redesign

It is important to note that the group process will be an activity network and in all probability will constitute more than a single activity. The presentation of this figure as a single activity matrix must not be taken to imply otherwise. There will be a number of activities, and just as the values of each node are conflated from their component actions (SINs), the final system specification will be conflated from all
the activities whose tools include interface elements, and that participate in Interface transactions.

4.23.1 A New Name - ATSA

Since the name “5S” was associated with a workplace organisation and standardisation framework (Osada 1991), the method developed in this chapter was renamed the Activity Theoretic Software Architecture (ATSA) method.

The word Architecture was chosen to reflect the level of abstraction the method operates at. Drawing some inspiration from the HCI literature, the system was envisaged as existing in, and being complimentary to, some information/activity space. An ATSA design was hoped to reflect consideration of how the users interacted with each other in such a space.

4.24 The ATSA Method As Built – A Stepwise Description

At the risk of repeating much of what has gone before, it is necessary at this stage to set out the ATSA method in a stepwise manner as a prescriptive framework. What follows is a statement of the ATSA method as constructed throughout this chapter. There are many aspects already justifying considerable future research, and doubtless other aspects may warrant modification or clarification under testing; but presented here is a sufficiently described AT based SAD method to be normatively tested in the following chapter.

The three phases of the ATSA method, with their component steps are as follows:

4.24.1 Elicitation Phase

Here the analyst extracts information from the group’s process in its raw state.

4.24.1.1 Identify Positions

The analyst must obtain a listing of all the positions in the stakeholder group. The objective is to identify all possible users of the system to be designed.

Many businesses may be expected to have some form of an organisation chart, and the positions may be read off from there. The analyst should consult with stakeholders to ensure accuracy and currency of the organisation chart. If an organisation has not yet charted its membership, the analyst may need to facilitate their doing so. Positions must closely correlate to group members as they appear in their hierarchic arrangement.

If there are multiple instances of a position, then the analyst may choose to consider simply one exemplar of each type. Careful note must be made, however, of all instances, as further analysis may reveal significant differences. Even should a position genuinely be duplicated, multiple instances must be recorded as the final system may require the ability to handle multiple simultaneous stations, multiple logon accounts and possibly multiple segregated data storage zones. The final system may require some form of overseeing function, assigned to a more senior user, and this might not have been indicated in the original organisation chart.
4.24.1.2 Identify Roles
The analyst must identify a listing of roles in the stakeholder group. Most business positions entail multiple separately identifiable duty-based roles; though not all business will have identified them in any formal way. Role information may be obtainable from documented position description information, retained by the stakeholders for personnel management purposes (such as recruiting).

Variations in role may be strong indicators that seemingly identical positions are in fact quite different. Should an analyst identify such variation, from role information, then they should reclassify the positions identified in the previous step, adopting some convenient and traceable nomenclature to differentiate them.

Ideally, roles should correlate closely to the subjects (protagonists) of activities. Unfortunately, a group’s role classifications are considerably more likely to be somewhat arbitrary duty-based classifiers, more indicative of responsibilities than of actual doings. Whilst it is very useful to gather this data, the Analyst must keep in mind that these roles may, ultimately, become less applicable or meaningful as the group’s agenda and process becomes recast into AT terms. It is well worth understanding to whom each position reports or answers, but to design a feasible system it will be far more important to know what resources they require to perform their work (from whom they obtain them, and under what conditions) and what resources then become available for others, as a result of their work.

If the stakeholder group does not have any form of role identification available, the analyst must attempt to elicit this through interview and observation. Rather than a setback, this situation is actually an excellent opportunity to directly elicit doing-based information rather than somewhat arbitrary duty (responsibility) based information. In this instance, roles identified by the analyst have the potential to remain current considerably further through the ATSA method, and may even survive with close correlation to stations.

4.24.1.3 Identify Candidate Instruments
The analyst must identify as many candidate instruments as possible from existing files, folders, forms, records, registers, lists, databases. These are data-like things (tangible or not) which roles require to perform their work, or which are made available for other roles as a result of their work. Candidate instruments may often exist as formalised document-types within the group.

Many groups may make use of specified formats for documents, such as templates or forms. It is important to clarify that these ‘template’ or pro-forma items should constitute separate (though related) candidate instruments to each instance or record that conforms to the template. To do the work of creating a record, for example, would require (among other things) the template as well as the information to be recorded. The template, in this case, will enter a doing as a resource but not then be available from that doing.

As the template (most typically) must remain available for repeated use, a copy of it may be considered to have been consumed, but the original will not have been destroyed. The source of a template is normally capable of supplying any required
(or permissible) number of copies, so long as organisational constraints are satisfied. These may be temporal, issues of template currency perhaps; or deontic, being issues of obligation, permission and authorisation etc. The updating or editing of the template itself is an entirely different doing, which will require the template, consume and (probably) destroy it, and produce a new template, available for consumption by other roles.

Some candidate instruments may appear to be containers for others, such as folders or files. The analyst should strive to avoid the temptation to define parent-child hierarchies between candidate instruments. There should only be resources and products. Upon closer examination, some potential container (meta) candidate Instruments could be subsumed into some constraint statement, imposing (for example) an obligation or procedure which labels, stores or restricts access to some set of candidate instruments. Other seemingly meta-level candidate instruments may be re-expressible as doings; the storage of, the labelling of, the retrieval of or the distribution of some other candidate instrument(s).

Under elicitation, the analyst may identify any number of intangible candidate instruments. These may take the form of procedural requests, permissions, state flags or conditions (such as date, time, cost, completeness or availability). Other intangible candidate instruments may be found in the tacit knowledge of skilled operators. The broad elicitation questions “what do you need to perform your task” and “what do you produce or provide to others” must be applied pursued thoroughly to elicit as many conditions, sequences, implicit or tacit data and the like as possible. Something as simple as a verbal call between workers may prove to be a crucial timing flag. Similarly, all such intangible resources (and products) must be recorded as they may need to be represented in some form as data objects, doings or constraint conditions in the final system.

The design of documents used within the group may have been mandated by organisational, legal or other external requirements. There may, therefore, be requirements for any virtualised (computerised) versions to retain particular design elements. Such constraints must be elicited and recorded by the analyst.

Regardless of any legalistic constraints, usability and user acceptance are often enhanced if the computerised forms of data objects, tools and procedures retain a resemblance to their previous real-world characteristics. The analyst must be sensitive to this effect, and be wary of any radical departures from current practice. Consultation with stakeholders is indicated.

The form of candidate instruments and the uses to which they are put (the purpose they serve) can inform the selection of onscreen interface widgets and switches. The analyst must be careful to allow this information to flow through into the interface specification ultimately passed to the interface designer.

**4.24.2 AT Analysis Phase**

Here the analyst recasts details of the group’s process into doing-oriented terms, towards assembling a truly activity oriented business process map.
4.24.2.1 Identify Goal Driven Actions

The analyst must now strive to identify actions. These are doings, conducted by a position (typically, acting in a role), which require some candidate instruments, which perform some work (under conditions and constraints, and to some purpose) and should then produce some candidate instruments for others to use.

As initially identified, these actions are rather short-lived. Under deeper analysis in later stages, they will be split up and later conflated into activities. The action at this stage therefore is a convenient loci for grouping the transactions of candidate instruments between roles. The action is not the fundamental unit of analysis under AT, the activity is, so this is a transitional stage. It is important for the analyst to remember that actions are identified largely as a means towards the identification of activities.

It is entirely permissible for each action to have multiple instruments entering and leaving; however there should be only one single protagonist (actor) which will correspond closely to a position in a particular role. Though it is not strictly necessary, the Actions may be represented as the nodes of a directed graph network, with INSTRUMENT transactions represented as arcs. Whether represented graphically or not, the analyst should feel free to reconfigure and rearrange the network iteratively, under consultation with the stakeholders, until all parties reach sufficient agreement that a realistic representation of the group’s process has been achieved. This step yields what is, in effect, a rough initial business process map (BPM).

More important, in many respects, is the identification of each action’s goals. Under the ATSA method, an action goal is a set of statements that detail what is done and why. They list the required components (candidate instruments incoming from other actions), the work that is done to, upon or with them, and the resulting candidate instruments. Any constraints and conditions must be faithfully recorded, especially those pertaining to the timing or sequencing of doings (temporal) or to do with obligation, permission or authorisation (deontic). These goal statements must be as clear and unambiguous as possible, as they are a kind of genetic information which will carry across to more refined interpretations of the network.

4.24.2.2 Decompose Actions to Single Instrument Nodes (SIN’s)

The analyst must now decompose the actions identified in the preceding stage, to Single Instrument Nodes (SINs). Each SIN will have only one single Instrument attached, either entering, or leaving.

It is crucial that each SIN inherits sufficient information from its parent Action’s Goal statements to explain what the instrument (received or sent) is for, to whom it is transacted, and its relationship to other SINs. Any temporal or deontic constraints must be inherited, and expanded upon to convey any constraints pertaining between the instruments involved with the parent Action.

Actions themselves may effectively disappear at this stage, as SINs are further utilised and possibly recombined in different ways to form activities. Later, to assist in clustering interface components by screen, it may prove necessary to identify the
parent actions from which SINs are drawn. It is thus also necessary to retain a record of SINs’ lineage so the identity and characteristics their parent actions can be reconstructed, as required.

As apparent inconsistencies and gaps arise, as is likely when a system is either undocumented or is being reinterpreted into a new conceptual framework, there may be a powerful temptation for the Analyst to indulge in re-design at this stage. Detailed piecewise deconstruction of Goal Directed Actions (GDAs) into SINs should go some way toward revealing duplications, inefficiencies and ambiguities; awareness of which can strongly provokes consideration of possible changes. Such apparent opportunities are more likely to be (or should be treated as) indicators of the need for further iterative consultation, elicitation and clarification. After all, it must be remembered that the extant group process does work in its current form to some extent, though of course there is always scope for improvement and the group itself will have recognised that or else they’d not have engaged an analyst to design a computerised facilitation system.

It is necessary to strive for a complete picture of the extant group process. The analyst should therefore, strive to avoid making design changes at this time, but should feel free to note any and all issues and record any ideas towards the upcoming redesign phase.

### 4.4.2.3 Conflate Activities

At this stage the analyst is able to finally identify the activities through the conflation of SINs. There are two factors by which to identify an activity; the commonality of subject (protagonist) and the logical consistency of goals under a single motive (object).

The analyst must firstly sort SIN’s according to their protagonist (actor). These are still associated with positions, and probably still closely associated with duty-oriented role descriptions. As these might no longer closely represent do-ers, the Analyst must be prepared to redefine roles as necessary, so long as it is the same individual user (position) performing as subject of the conflated activity.

The Analyst must consider the goal statements of the SINs and sort for those which could be logically collected under a consistent motive statement. Until some future refinement of the ATSA method incorporated a rigorously formal multi-modal logical consistency checking technique, this will require some careful consideration on the part of the analyst, who would most probably benefit from consultation with the stakeholder.

Conflation by goal consistency may give rise to further redefinition or clarification of Roles. The analyst must be prepared to conduct these two sorting processes iteratively, as required. Conflated activities may closely resemble the initially identified actions, but there is no guarantee of this. The analyst must be prepared for considerable variance from the pattern of initial actions. Should the final activity network closely resemble the initially proposed actions, then the analyst has still achieved both a confirmation and the assemblage of logical constraints which accurately model the instrument transactions of the process.
Chapter 4 – Method Building

It is vital that the conflated motive for each identified activity *inherits* all the constraints and conditions of its component SINs in a consistent manner. Any conflated activity must contain full details of its doings, of all required Instruments and of all transformations and products.

The second-sort requirement that SIN goals be logically consistent provides an upper bound, identifying all of the common-subject SINs which *could* possibly exist under a single Activity. It is anticipated however that it will frequently not be the case that all such SINs will constitute a single activity. A lower bound for the number of SINs needed to conflate an activity might be expressed as coherence. A sufficient number of consistent goal statements will need to be conflated to form a coherent activity motive statement. This is an imprecise, value-laden exercise on the part of the analyst. Future research work will be needed to identify further heuristics and guidelines to inform this concept.

The set of activities conflated can be illustrated as a directed graph of the activity network. Each node will now be a fully formed activity which could be described according to its Engström matrix elements. Directed arcs represent instrument transactions. The adjacency matrix of this digraph is the Combined Activity Table (CAT).

The self-correcting nature of the CAT, which requires for its completeness full details of all arc transactions, serves as a prompt for the analyst to ensure all necessary data has been elicited, deduced or conflated.

The resultant network and its CAT constitute a refined activity oriented BPM. This is a significant outcome in its own right, serving as a useful business analysis tool. An AT oriented BPM is, by definition oriented by doing (processes) and as such is of considerable use for any business seeking accreditation under the ISO:9001 standard of quality management.

The BPM provides a mutually comprehensible tool for the stakeholder and Analyst to examine the entire group process. It facilitates arguing the case for designing a central enabling computer system. There should be something of a potential void near the centre of the network where such a System *could* be. Inefficiencies, duplications, multiple handlings and redundancies in the current process should be highlighted and possible solutions discussed.

The ATSA method in full, assumes the stakeholder would authorise redesign of the process, incorporating a facilitating computer system.

4.24.3 Re-Design Phase

Here the analyst now takes the part of a designer. Although traditionally the project might be handed off to a designer at this stage, ATSA allows for the entire SAD to occur under one consistent paradigm, and strongly advocates unifying analysis with design. Under consultation with the stakeholder, the designer seeks to reconfigure and rearrange the group’s process to enhance efficiency through the agency of a central facilitating computer system. Once the stakeholders are accepting of a design, the method will describe it in sufficient detail to specify its requirements for detailed software-code and interface construction and implementation.
The result of this third phase is a redesigned BPM, now featuring a specification for a central facilitating computer system. The system design will be well suited to the users’ doings which should positively effect usability, user acceptance and thus success. With consultation between the analyst-designer and stakeholders, the arrangements of Individuals in positions and roles may have changed, but this should only result in a closer match of the system to the group’s agenda.

The system will be specified in terms of its instruments and their transactions (analogous to data in, data out and transforms) and the clustering of its interfaces according to users’ doings.

4.24.3.1 Run Node Reduction Heuristics

If the stakeholder agrees inefficiencies, duplications, multiple handlings and redundancies exist in the current process then a redesign of the process is indicated. This may involve redefining numerous roles and reassigning tasks (Instrument transaction flows). It is strongly recommended that the designer retain the structure of positions (people) unless the stakeholder specifically requires changes at that level.

Irrespective of changes to which doings are performed by whom, the BPM should be rationalised and modified to represent the nature and behaviour of the System.

The designer should identify pipes, which are those arcs (instrument transactions) between activities which have a common subject (owner) whereby the subject sends something to themselves; and then joints which are nodes (activities) through which an instrument passes unchanged.

Locations on the activity network (BPM) where pipes and joints coincide are strong candidates for collapse and or redesign. The designer must pay particular attention to the elicited constraints (purpose, sequence, obligation etc) to ensure any modification is not in breach of the agenda.

Where necessary, activities may be split, joined or created. Whenever this occurs, their component SINs must be considered. As in previous stages, though SINs may be moved, duplicated or deleted as required, their genetic identities must be retained.

A space between the activities where some facilitating computer system could be should become clear. Many (hopefully, most) activities could be seen to be tangential to this system, able to exchange their candidate instruments through or with it.

4.24.3.2 Bend Instrument Paths to System

As the location and boundary of the potential system becomes apparent, the designer should actively seek to reroute (bend) the instrument transactions through it. The notion is to consider that Instruments reside inside the System’s Data Repository (SDR). Activities around the system boundary will exchange data with the System rather than directly with each other. If activity A passes instrument I to activity B, then in reality, A updates the value of I in the SDR, and B reads this value from the SDR, as required.
Following this process, some activities may be seen not to conduct any transactions across the system boundary. These will be deemed peripheral activities and cease to be directly any part of the system. Details of peripheral activities should be retained however, to provide useful context and to facilitate any future revision or redesign.

Objects which are stored in the SDR are deemed to be true instruments, as distinct from the candidate instruments which have been discussed up until this stage. These closely correspond to data objects and variables in the final system build, but the ATSA method deliberately tries to avoid such labels in order to retain paradigm neutrality. ATSA should strive to supply adequate, feasible, useful and acceptable high level specifications of the system and its interface, such that the stakeholders may comprehend what they are purchasing, the users may comprehend what they’ll be doing and the builders may apply whichever techniques and languages they deem best.

**4.24.3.3 Convert Temporal Rules into Deontic DivLABs**

Following any modifications from the preceding stages, Activities should retain a complete suite of constraints, conflated from the inherited characteristics of their component SINs. The rules of these SINs consist largely of temporal (sequential) constraints of the general form ‘must do during state S’. The designer must now strive to rewrite as many of these rules as possible into the more deontic flavoured DivLAB style. In general, DivLAB statements should take a general form such as ‘run iff Instrument ‘state S’ = TRUE’. The DivLAB statements constitute a listing of prerequisite ‘ingredient’ Instruments and their required states or values.

DivLAB constraints are, in effect, gatekeeper functions, preventing the execution of a doing until required conditions are satisfied. This places an obligation upon the provider of an instrument (the system, in every case) to maintain and monitor their state. Rule conditions however can oblige the consumer of an instrument to ensure the state of instruments. The degree to which rules are converted into DivLAB's is an indication of the reliability of the final system, as DivLABs may be built directly into the functionality of the system.

Observe that, in this process, SINs may be profoundly modified. Afterwards, many of SINs occurring within activities will take the form of receivers. New SINs may thus be created, and many will be rewritten. Care must be taken to ensure accurate inheritance (under conversion or translation, as required) of SIN characteristics, conditions and constraints.

**4.24.3.4 Convert Goal Statements to Pseudocode**

The designer must now formalise and standardise the goal statements within each activity. These must be expressed as succinctly and accurately as possible. It is recommended that a reduced vocabulary form of structured language be employed, perhaps a form of generic pseudocode.

The object (motive) of each activity will then be a logically consistent, highly structured, set of goal statements in a form easily passed to a coder as functional requirements.
4.24.3.5 Instrument List

There is value in sorting the specification by its instrument rather than by its activities. The coder, especially one who follows an object-oriented paradigm, is more likely to grasp the design on a per-data-object (variable) basis.

A database of some kind could prove a useful way to present this information, as it would allow the design to be sorted in various ways, and would retain activity information with each instrument. Each instrument record would offer a meaningful name and some classification of its functionality defined by the transforms it must undergo and the uses to which it is put as a resource.

It is crucial that each instrument’s transactions are easily traceable through the activity network. It is important to wire up the transactions correctly, clustering Instrument handling interface widgets according to station.

4.24.3.6 Identify Switches and Combine them into Screens etc

Whilst the functions of the system are described through the instrument list and goals statements (indicating what data objects must move in and out of the system, under what transforms and constraints), the specified nature of the interface must also be generated.

Each SIN will handle a single one terminus of an instrument transaction. These will cross the system boundary at the interface. The ATSA method should provide an interface designer with a sufficiently detailed description of who is handling what, and to what end, that (according to the builders adopted paradigm) a selection of screen widgets (switches) becomes apparent. Once again, the ATSA method seeks to supply high-level paradigm neutral specification, to support a wide range of options for the builder.

Whilst much has been written, under various paradigms, about the selection of widgets and colour schemes etc, an important element of interface construction that is not well understood in the literature [Dix et al. 2004] is the clustering of switches for best use. It is recognised that the assemblage of widgets can greatly impact usability, acceptance and thus success of a system.

Under the ATSA method, interface stations are assembled according to the coincidence of activity subjects (protagonists) in individual users (positions). Each station constitutes one or more screen-sets, one per activity. That is to say, the interfaces are summoned on a per activity basis. There should be no need for the user to be clumsily swapping between modes or usage instances if the interfaces have been clustered according to what they need to do.

Each activity’s screen-set may contain a number of screens (windows, instances, tabs etc). Each of these should correlate to an action. actions are no longer clearly defined under the method at this stage however, but may be understood to be sets of SINs (sub sets of activity) clustered according to goal. Some indicator of the likely location of actions may be obtained from the genetic lineage of each SIN. It must be noted that action clustering may have changed as the BPM was redesigned to accommodate the facilitating system. It is at the level of screen clustering that ATSA still retains the greatest flexible capacity for the builders’ personal intuition. More
rigorous techniques for reading off actions (screens) from the specification may emerge under future research, perhaps drawing upon logical analysis of deontic and temporal constraints.

4.25 Conclusion

There are a number of opportunities within the ATSA method as it stands at the conclusion of this chapter, for refinement of technique and process. The use of some database-like tool to assist the analyst/designer is strongly indicated. The utility of adopting multimodal logical operators and rigorous consistency checking rules is indicated. Such explorations lie outside the scope of this research project but strongly suggest future research directions.

Having built the ATSA method both in concept and in sufficient detail to allow initial testing, then according to the normative methodology & in accordance with AT, it is now necessary to test its feasibility in a genuine case. In the conduct of this test, no further changes to the method as detailed above are permitted. The objective will be to test, not so much the ATSA method itself, but the research hypothesis;

Systems Analysis and Design (SAD) may be improved if conducted through a prescriptive but agnostic method, constructed according to Activity Theory (AT) principles.

Should the normative, indicative test of the ATSA method in its current form demonstrate feasibility, further research along the lines indicated would be justified, towards a genuinely new commercially viable SAD.
Chapter 5 – Method Evaluation

5.1 Introduction
In the methodology discussion (chapter 3), a normative research framework was selected under which to achieve the four stated research goals. Having identified some challenges in existing Systems Design and Analysis (SAD) methods and hypothesised some means of improvement to address the challenges, chapter 4 detailed the construction of a SAD method, based on that hypothesis. In this chapter it is necessary to address the fourth research goal by evaluating the new method by applying it against a realistic case study.

As explained in chapter 4, the first two stages have been satisfied by the literature review (chapter 2). This chapter therefore addresses the third stage, the synthesis of a proposed improvement or under the March and Smith (1995) terminology, the build phase.

As previously set out section 3.5.4, in the method evaluation, it will be necessary to establish that the novel method can do what a SAD method does. The novel method should therefore produce essentially the same result, being in this case according to Pressman (2001), sufficient information regarding the inputs, outputs and transformations of data within a system to allow for that system to be built.

Chapter 4 presented a detailed account of the building of a novel SAD method based upon the precepts of Activity Theory (AT). Table 5.1 (below) gives a summary of the major steps in the ATSA method, as described in greater detail in section 4.24.

<table>
<thead>
<tr>
<th>Elicitation Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify Positions</td>
</tr>
<tr>
<td>Identify Roles</td>
</tr>
<tr>
<td>Identify Candidate Instruments</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AT Analysis Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identify Goal Driven Actions</td>
</tr>
<tr>
<td>Decompose Actions to Single Instrument Nodes (SIN's)</td>
</tr>
<tr>
<td>Conflate Activities</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Re-Design Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run Node Reduction Heuristics</td>
</tr>
<tr>
<td>Bend Instrument Paths to System</td>
</tr>
<tr>
<td>Convert Rules</td>
</tr>
<tr>
<td>Convert Goal Statements to Pseudocode</td>
</tr>
<tr>
<td>Instrument List</td>
</tr>
<tr>
<td>Identify Switches and Combine them into Screens etc.</td>
</tr>
</tbody>
</table>

Table 5.1: The ATSA Method, Stepwise

In the following sections of this chapter, the ATSA method will be applied, without further development of its processes from those described in chapter 4, to a real-world case. The objective is to test if indeed it shows the potential to produce essentially the same result as a SAD method.

5.2 Choice of Case Study
The ATSA method was primarily intended to facilitate SAD for a multi-user business scenario, where a coherent group conduct doings towards a common
agenda. To establish that the ATSA method can analyse such a case and produce a system specification, and remain a feasible study within the scope of this research project, it was necessary to investigate a real-world business which had an identifiable agenda but was just large and complex enough to demonstrate the method’s workability.

The researcher identified and investigated the operations of a particular small business and became familiar with its personnel and processes. For reasons of commercial confidentiality in a competitive industry however, neither the business identity, that of its stakeholders nor the industry specific nature of its operations may be revealed. The business’ functions were sufficiently generic however as to represent typical commercial operations of supply, demand, the satisfaction of customer needs and job tracking.

5.2.1 A Specialist Labour Hire Company

The test case chosen was a small and business in only its second year of operation, which supplied industry specific labour on a per-job basis. To maintain commercial confidentiality, this group will henceforth be referred to simply as ‘the business’.

The business employs only a small number of administrative staff, but retains the services of qualified contract labourers. These labourers, who hold independently obtained industry-specific qualifications, are assigned work shifts by the business.

5.3 Elicitation Phase

According to the ATSA method, as built in the previous chapter and defined, stepwise, in section 4.24, the primary function of this phase is to gather information about the group and their doings, prior to translating these into AT terms.

The first step is to identify the involved individual group members, in terms of their organisational positions.

5.3.1 Identify Positions

The business retained five full-time administration staff but only had an informal description of positions. Each staff member seemed to have a fairly clear and mutually understood place within the structure, but no fixed position titles had been documented.

Initially the two most senior group members (joint owner-operators of the business) both selected the duplicate title ‘Managing Director’. The ATSA method directs that such duplication should at least be carefully recorded, and ideally, resolved by differentiating work functions where possible. Under further elicitation, it was realised that one individual was based largely in the office, overseeing financial and business administration, whilst the other was frequently out attending work sites and overseeing the duties of the contracted specialist operatives. The Analyst therefore suggested the more specific position titles, Managing Director Administration and Managing Director Operations.

Under further consultation with each staff member and the senior-most stakeholders, the following position titles were generated:
5.3.2 Identify Roles

As a small company, the number of permanent staff was quite small. The business anticipated expanding the number of permanent staff, and thus positions, as the number of clients and the breadth of work grew. It was expected that duties and responsibilities would be distributed between an expanded set of positions.

The ATSA method requires that positions be decomposed into roles, towards identification of activity subjects. The anticipated restructuring of the business under expected future growth only strengthened the case for identifying roles. Existing, and future positions could be relatively quickly defined as sets, or of these roles, or variations upon them.

With the assistance of the analyst, the business identified a number of roles. As anticipated by the ATSA method, the stakeholders described roles according to duty and responsibility. These were each assigned to specific positions but several duties were shared to greater or lesser extents by more than one individual. The directors, for example, could act in any capacity, and the Administrative Assistant could be delegated to perform numerous tasks. This vagueness was symptomatic of the small number of staff, and was expected to largely disappear as new, more specific, positions were created.

As duties were not precisely mapped to individuals, but could be defined quite clearly in themselves, these roles presented a more meaningful locus for analysis of the business’ doings, as anticipated by the ATSA method. From this point, doings could be more meaningfully described in terms of their protagonist roles. It was important, however, to record both the assignment of roles to positions, and the crossovers of duties, so as to correctly cluster system interfaces and functions to individual Stations in the final phases of the method. Again the ATSA method, as described, closely suited the real world experience.

The business chose to assign a three tier hierarchy of authority to roles. Purely functional roles were designated ‘Administrative’. Decision making roles were assigned the status of ‘Officer’. Final authority and corporate responsibility was vested solely in the owner operators, and their roles were designated as ‘Directorial’. The ATSA method, as described, did not specifically anticipate a differentiation of roles by authority (a notion of organisational rank). It was hoped that the ATSA method’s recording of deontic relationships would capture as much detail as required.

The roles defined within the business’ structure are set out in Table 5.3, below.
Chapter 5 – Method Evaluation

<table>
<thead>
<tr>
<th>Role</th>
<th>Title</th>
<th>Directorial</th>
<th>Officer</th>
<th>Administrative</th>
</tr>
</thead>
<tbody>
<tr>
<td>(CFO)</td>
<td>Chief Finance Officer</td>
<td>(MD-Admin)</td>
<td>Managing Director Administration</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>oversees all financial transactions for the Business. Retains sole authority and control over all financial resources and transactions. Uses the MYOB accounting package and liaises with external accountancy services. Note: MYOB is a widely used proprietary 3rd party Computer Application.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(PM)</td>
<td>Pay Master</td>
<td>(MD-Admin)</td>
<td>Managing Director Administration</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>collects, collates and confirms a timesheets and generates payments to employees through the MYOB accountancy package and (typically) through direct internet bank transfers. This person also monitors taxation, superannuation and all other associated details. Subsidiary function within the duties of the (CFO) Chief Finance Officer.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(RTW)</td>
<td>Workplace Contact for Return to Work Program</td>
<td>(MD-Admin)</td>
<td>Managing Director Administration</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>the designated person that initiates and co-ordinates appropriate WorkCover and return-to-work functions. Liaises where necessary with WorkCover, insurance agents, other external agencies and/or specialised return-to-work co-ordinator(s). Employees are advised to contact this person initially for all issues involving worksite injuries, WorkCover enquiries and all return-to-work matters. Note: WorkCover is a New South Wales Government statutory authority.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(CSA)</td>
<td>Chief Safety Auditor</td>
<td>(MD-Opns)</td>
<td>Managing Director Operations</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>conducts and co-ordinates worksite audits and onsite employee performance reviews.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(EL)</td>
<td>Employee Liaison</td>
<td>(MD-Opns)</td>
<td>Managing Director Operations</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>directly assists onsite employees with their concerns. This person provides supervision, mentoring and co-ordination. Reviews and audits worksite paperwork from employees.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(SLO)</td>
<td>Strategic Liaison Officer</td>
<td>(SM)</td>
<td>Strategic Manager</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>co-ordinates, advises and assists the MDs with long-term corporate planning issues, the formulation of policies and procedures and in dealings with: statutory &amp; regulatory bodies; government agencies; external operators; external corporations; oversight and advise on all official communications leaving the office.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(SMPO)</td>
<td>Safety Management Planning Officer</td>
<td>(SM)</td>
<td>Strategic Manager</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>drafts, oversees, maintains and manages the Safety Management Plan (SMP) including all corporate Policies and Procedures.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(PolA)</td>
<td>Policy Administrator</td>
<td>(SM)</td>
<td>Strategic Manager</td>
<td>(SM) Strategic Manager</td>
</tr>
<tr>
<td></td>
<td>drafts, oversees, maintains and manages all Policies and Procedures. Note: Policies must be approved and signed by an MD, and all copies managed by the DCO</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(QSO) Quality Systems Officer</td>
<td>Officer</td>
<td>(SM) Strategic Manager</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------</td>
<td>-----------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drafts, oversees, maintains and manages the Quality Management System (QMS).</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(PRO) Public Relations Officer</th>
<th>Officer</th>
<th>(SM) Strategic Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advises, assists and co-ordinates the MDs and other officers in matters relating to: corporate image; all forms of media advertising; media or public announcements; Design and booking of media releases, advertisements etc.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(ITSO) IT Support Officer</th>
<th>Officer</th>
<th>(SM) Strategic Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advises assists and coordinates the procurement, maintenance and operation of Information Technology equipment, inclusive of computers, printers, routers etc. Consumables, such as printer ink and toner may also be managed by the (OffA) Office Supplies Administrator. All expenditures on IT capital equipment, consumables, services and subscriptions may only be authorised by the (CFO) Chief Finance Officer.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(WebA) Web Site Administrator</th>
<th>Administrative</th>
<th>(SM) Strategic Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duties: This function is subsidiary to the Public Relations and IT Support Officers. This person maintains the corporate web site, adding or modifying content and presentation as directed.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(MO) Marketing Officer</th>
<th>Officer</th>
<th>(OM) Office Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liaises with potential Client Companies. Offering services following opportunities identified in Tender announcements, industry media, word-of-mouth etc. Note: All external representations of Business services are co-ordinated with the (PRO) Public Relations officer and (SLO) Strategic Liaison Officer, and approved by an MD.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(CLO) Client Liaison Officer</th>
<th>Officer</th>
<th>(OM) Office Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liaises with Client Companies: checking the Scope of Work bookings; checking job Commencement and/or Completion; checking Customer satisfaction</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(JobA) Job Booking Administrator</th>
<th>Administrative</th>
<th>(OM) Office Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Takes enquiries from Clients and books the details of requested jobs.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Chapter 5 – Method Evaluation

<table>
<thead>
<tr>
<th>(EmpA)</th>
<th>Employee Placement Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative</td>
<td>(OM) Office Manager</td>
</tr>
<tr>
<td></td>
<td><em>May also be performed by the MD-Opns, or the AA (under supervision)</em></td>
</tr>
</tbody>
</table>

Allocates employees to specific jobs, being careful to meet all operational and administrative requirements.

<table>
<thead>
<tr>
<th>(DCO)</th>
<th>Document Control Officer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Officer</td>
<td>(OM) Office Manager</td>
</tr>
</tbody>
</table>

Maintains records of corporate files and documents, including Policy and Procedural documents, tracking currency, availability, completeness and version control. The DCO also maintains and manages copies of external information resources, such as industry relevant Standards, Legislation and Regulations.

<table>
<thead>
<tr>
<th>(ICO)</th>
<th>Internal Communications Officer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Officer</td>
<td>(OM) Office Manager</td>
</tr>
<tr>
<td></td>
<td><em>NB: may also be conducted by the MD-Opns, or the AA</em></td>
</tr>
</tbody>
</table>

Co-ordinates internal distribution of information including: employee newsletter; regular OHS reports and advisory post outs; emergency medical contacts; onsite work papers; policy and procedural updates; logging corporate correspondence.

<table>
<thead>
<tr>
<th>(OffA)</th>
<th>Office Supplies Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative</td>
<td>(OM) Office Manager</td>
</tr>
</tbody>
</table>

Duties: This person tracks supplies of stationery and maintains stocks of necessary materials, inks and similar materials. All procurements must be approved by the Chief Finance Officer.

<table>
<thead>
<tr>
<th>(OHS)</th>
<th>OH&amp;S Co-Ordinator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Officer</td>
<td>(OM) Office Manager</td>
</tr>
</tbody>
</table>

Overssees OH&S inspections and compliance within the corporation, most especially within the office environment. Chair meetings of the OHS committee and write corporate OHS reports, This person will hold, or be studying towards, an appropriate OH&S certification.

*Note: ‘OH&S’ (Occupational Health and Safety) refers to New South Wales Legislative obligations upon all employers to maintain safe working environments.*

<table>
<thead>
<tr>
<th>(PA)</th>
<th>Directors’ Personal Assistant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative</td>
<td>(AA) Administrative Assistant</td>
</tr>
</tbody>
</table>

Duties: This person provides general clerical and administrative support the Managing Directors including: maintaining meeting schedule and diary; maintaining external contact details; taking minutes of business meetings.

<table>
<thead>
<tr>
<th>(GenA)</th>
<th>General Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrative</td>
<td>(AA) Administrative Assistant</td>
</tr>
</tbody>
</table>

Day-to-day administrative office functions: retrieving, filing and updating records, facilitating and conducting routine communications, etc – generally assisting other officers, managers and directors in the performance of their duties, as required.
### Table 5.3: Roles

<table>
<thead>
<tr>
<th>(HumA) Human Resources Administrator</th>
<th>Administrative (AA) Administrative Assistant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duties: This person co-ordinates administration of employees. This includes: recruitment advertising; receipt and processing of employment applications; applicant reference checks; tracking employee documentation &amp; CoC’s etc; monitoring the “employee status” system; collating and checking employee time sheets</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(FtgA) Fatigue Administrator</th>
<th>Administrative (AA) Administrative Assistant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operates the FAID software package to monitor the fatigue records for employees – reporting directly to the MD,Adm.</td>
<td></td>
</tr>
<tr>
<td>Note: FAID is a proprietary 3rd party Computer Application.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(EqpA) Equipment Administrator</th>
<th>Administrative (AA) Administrative Assistant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performs the basic recording and maintenance of records of industry-specific work equipment stored at the Business premises.</td>
<td></td>
</tr>
</tbody>
</table>

### 5.3.2.1 AN ASIDE: Selecting Business Procedures for Analysis

In accordance with the methodological choice to evaluate the proposed method via a normative case study, as described in section 3.5.4, only a sufficiently representative sample analysis would be required to demonstrate the feasibility of the proposed method.

At this point therefore, it was necessary to briefly set aside the stepwise trialling of the ATSA method, in order to select business processes for deeper analysis. The business, though small, conducted a number of identified processes, and numerous as yet undescribed subordinate processes, and it would have been well beyond the scope and needs of this method evaluation to conduct an exhaustive analysis of them all.

The business’ core operations involved the short-term per-project matching of casual skilled employees to specific client projects. During the period of investigation, the business commenced preliminary identification and classification of the primary business procedures subordinate to that core function.

At the time of the analysis, the business was in the process of attempting to catalogue its internal process with a view towards gaining ISO:9001 accreditation. During the elicitation phase, sixteen processes had been identified but not all of them had been fully documented. As might be expected in most businesses, none of them were specifically described in AT terms.

A brief listing of the identified processes is presented in Table 5.4 (below), together with an explanation of the choice of the processes used for the remainder of the normative evaluation exercise. An anonymised version of the business internal
Chapter 5 – Method Evaluation

documentation as it stood at the time of elicitation is presented for the sake of completeness in Appendices B, C and D.

<table>
<thead>
<tr>
<th>Business Procedure</th>
<th>Selected</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advertising for Employees</td>
<td>No</td>
<td>Largely external to any new System.</td>
</tr>
<tr>
<td>Response to Advertising</td>
<td>No</td>
<td>Too small and too manual.</td>
</tr>
<tr>
<td>New Employee Application</td>
<td>Yes</td>
<td>Established the point of origin for numerous important candidate instruments used in later processes.</td>
</tr>
<tr>
<td>Marketing procedures</td>
<td>Yes</td>
<td>A subset is examined, where Jobs requests are received. This subset is subsumed into the Job Booking procedure.</td>
</tr>
<tr>
<td>New Client Establishment Procedure</td>
<td>Yes</td>
<td>Client details needed in System.</td>
</tr>
<tr>
<td>General Job Booking Procedure</td>
<td>Yes</td>
<td>A core business activity.</td>
</tr>
<tr>
<td>Job Booking: Job Scope Procedure</td>
<td>Yes</td>
<td>Subsumed into the Job Booking procedure.</td>
</tr>
<tr>
<td>Job Booking: Employee Selection Procedure</td>
<td>Yes</td>
<td>A core business activity.</td>
</tr>
<tr>
<td>Job Booking: Confirm Employee Procedure</td>
<td>Yes</td>
<td>Subsumed into the Employee Selection procedure</td>
</tr>
<tr>
<td>Job Booking: Job Confirmation Procedure</td>
<td>Yes</td>
<td>Subsumed into the Job Booking procedure.</td>
</tr>
<tr>
<td>Job Check Form Procedure</td>
<td>Yes</td>
<td>A core business activity.</td>
</tr>
<tr>
<td>Customer Satisfaction Procedure</td>
<td>No</td>
<td>Largely undocumented at the time of elicitation. Some elements subsumed into the Employee Experience Register process.</td>
</tr>
<tr>
<td>Job Progress Check Procedure</td>
<td>No</td>
<td>Too small and too manual.</td>
</tr>
<tr>
<td>Employee Experience Register</td>
<td>Yes</td>
<td>A core business activity.</td>
</tr>
<tr>
<td>Exceptions: Occurrence Procedure</td>
<td>No</td>
<td>Some exception cases, which were relatively detailed and constituted check points in core processes, were subsumed.</td>
</tr>
<tr>
<td>Corrective Action Report (CAR) Procedure</td>
<td>No</td>
<td>Largely undocumented at the time of elicitation.</td>
</tr>
<tr>
<td>Bill Client Procedure</td>
<td>No</td>
<td>But included in early analysis to track the end point of some Candidate instruments.</td>
</tr>
<tr>
<td>Payment Processing Procedure</td>
<td>No</td>
<td>But included in early analysis to track the end point of some Candidate instruments.</td>
</tr>
<tr>
<td>Employee Pays Procedure</td>
<td>No</td>
<td>Largely undocumented at the time of elicitation.</td>
</tr>
</tbody>
</table>

Table 5.4: Business Procedure Selection
Some procedures were insufficiently documented to explore further without extensive research beyond the scope of this method evaluation. Others were almost entirely conducted outside of the core office processes and would thus have little direct engagement with any bespoke computerised facilitation system.

The final selection constituted the following six processes: New Client, New Employee, Job Booking, Job Status, Employee Selection and Review Employee.

### 5.3.3 Identify Candidate Instruments

The business procedures were an excellent starting point for identifying Goal Driven Actions (GDAs), in which candidate instruments were transacted between roles. The ATSA method requires however, that candidate instruments be identified first.

The analyst elicited a list of documents in use by the business in the identified processes. It was hoped that these would serve as a solid basis for suggesting candidate instruments.

Some of these documents were individual records, for specific employees, clients or jobs etc; some were templates, lists or forms used to elicit or store such records. Other documents were communications, usually from the business to individuals or groups.

Table 5.5 (below) presents a list of documents identified by the analyst with brief descriptions. Each is identified as a record, list, form or as some other form of document, such as a communiqué.

<table>
<thead>
<tr>
<th>Candidate Instrument #</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI 1</td>
<td>client application form</td>
<td>Used to elicit and record details of companies wishing to use the Business’ services</td>
</tr>
<tr>
<td>CI 2</td>
<td>client file</td>
<td>A record for each client which has engaged the services of the Business. Records contact details, account details etc.</td>
</tr>
<tr>
<td>CI 3</td>
<td>employment application</td>
<td>document(s) submitted by person seeking employment</td>
</tr>
<tr>
<td>CI 4</td>
<td>employee file</td>
<td>A record for each Business employee, especially those casual individuals whose qualified services the Business supplies on a per job basis to its clients.</td>
</tr>
<tr>
<td>CI 5</td>
<td>employee folder</td>
<td>the collection of employee files</td>
</tr>
<tr>
<td>CI 6</td>
<td>quick reference board</td>
<td>An overview listing of employees – showing Name, Qualifications, home location and “colour-code” status (Green = safe to use, Amber = use under supervision/review, Red = do not use)</td>
</tr>
<tr>
<td>CI 7</td>
<td>employee suitability form</td>
<td>A form used to collect and record data to assess employee suitability; including referee details and comments, work history, work scenario specific experience etc.</td>
</tr>
<tr>
<td>CI 8</td>
<td>Employment Information Pack</td>
<td>A collection of standardised documents informing a new employee of their rights, duties, obligations and the terms and conditions of their employment.</td>
</tr>
<tr>
<td>CI 9</td>
<td>employment enquiries folder</td>
<td>Keeps a record of all employment applications, successful or otherwise</td>
</tr>
<tr>
<td>CI 10</td>
<td>acceptance letter</td>
<td>A form letter advising an applicant of a successful application for employment.</td>
</tr>
<tr>
<td>CI 11</td>
<td>rejection letter</td>
<td>A form letter advising an applicant of an unsuccessful application for employment.</td>
</tr>
<tr>
<td>CI 12</td>
<td>Timesheets</td>
<td>A standard form used by employees to record their periods of employment. They must be authorised by the client and may be reviewed by a Business Officer and subjected to audit. These are used to calculate weekly employee pays, when checked against the previously set rates of pay and duties. The Timesheet also records special allowances such as travel or accommodations.</td>
</tr>
<tr>
<td>CI 13</td>
<td>site maps</td>
<td>Job specific maps extracted from online and other sources. Supplied to employees to apprise them of site locations, access and conditions.</td>
</tr>
<tr>
<td>CI 14</td>
<td>suitability checklist</td>
<td>A worksheet used to confirm all stages of employee selection on a per-job basis</td>
</tr>
<tr>
<td>CI 15</td>
<td>rates schedule</td>
<td>A listing of charge rates applied to clients for jobs. This list is determined by the CFO.</td>
</tr>
<tr>
<td>CI 16</td>
<td>pay schedule</td>
<td>A listing of pay rates given to employees – according to qualification, work type etc.</td>
</tr>
<tr>
<td>CI 17</td>
<td>bonus schedule</td>
<td>A listing on bonus pays given to employees for such things as travel Allowance or Living Away from Home Allowance (LAHA)</td>
</tr>
<tr>
<td>CI 18</td>
<td>job booking sheet</td>
<td>A full record for a given job. Often uses an Invoice Number (CI 28) as a unique Job identifier (<em>NOTE: an immature Business process at the time of analysis</em>). Job Booking Sheet also constitutes a form of Worksheet for documenting a Job, and gives rise to a Job Sheet (CI 22) and Job Letter (CI 21) etc.</td>
</tr>
<tr>
<td>CI 19</td>
<td>job scope enquiry form</td>
<td>worksheet used to elicit details of a specific jobs nature. Contains a listing of job types – also used in the employee experience register.</td>
</tr>
<tr>
<td>CI 20</td>
<td>Job scope report</td>
<td>The record of job type, location and other pertinent details for a specific job.</td>
</tr>
<tr>
<td>CI 21</td>
<td>job letter</td>
<td>Only generated at the END of job booking and employee selection. It constitutes a per-job, per-employee contract issued to confirm engagement for a specific job. It sets out the location, the duties, period and hours of employment, pay rates, allowances and deductions and also details any special conditions pertaining.</td>
</tr>
</tbody>
</table>
| CI 22 | job sheet | overview-record of a job, pinned to the Pending Job Board or the Current Job Board.  

*NOTE: the Job Sheet is like a TOKEN which represents the total information package associated with a given job.* |
| CI 23 | pending job board | Bulletin board where Job sheets are displayed. These are Job Sheets for jobs agreed and staffed, but not yet occurring.  
When jobs commence, they’re moved to the current job board  

*NOTE: the job board is like a FLAG, which marks the status of a job as pending or current – but also serves as an overview display of all items with that status*. |
| CI 24 | current job board | Bulletin board where Job sheets are displayed. These are Job Sheets for jobs currently occurring.  
After completion, Job Sheets are removed and actioned for involving etc.  

*NOTE: the job board is like a FLAG, which marks the status of a job as pending or current – but also serves as an overview display of all items with that status*. |
| CI 25 | FAID® | 3rd party proprietary software used to calculate levels of fatigue produced from specific schedules of work and travel etc. |
| CI 26 | MYOB® | 3rd party software for keeping account of the Business finances. Used to generate invoice numbers, print invoices, purchase orders & generate pay slips etc. |
| CI 27 | pay sheet | Physical printout generated by the MYOB® system. Issued each week to each employee, detailing work hours, pay, special allowances and deductions for taxation, superannuation etc. |
| CI 28 | invoice number | Generated by the 3rd party MYOB® software package.  

*NOTE: sometimes used as a unique job number*. |
| CI 29 | invoice | Printed by the 3rd party MYOB® software package and posted to the client. |
| CI 30 | Online Banking | Web-based 3rd party online system for accessing the Business’ bank account(s) for balance enquiry, deposit, withdrawal, bill payments etc. |
| CI 31 | pre work briefing | A legally required verbal safety address to all job personnel prior to the start of each days work, conducted by the senior employee onsite.  
A signed attendance ROLL is taken and recorded, then forwarded to the Business office for archiving. |
| CI 32 | worksite protection plan | Detailed legally required worksite safety plan prepared by senior employee prior to the start of each day’s work at a job.  
Recorded on standard pre-printed FORM and forwarded to the Business office for archiving. |
| CI 33 | employee experience register | A record of individual employee’s work experience, used to assist selection of best employees for each job.  

*NOTE: not yet clearly defined nor implemented*. |
<table>
<thead>
<tr>
<th>CI</th>
<th>Business Document</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CI 34</td>
<td>client review</td>
<td>Worksheet used to elicit client feedback on completed jobs</td>
</tr>
<tr>
<td>CI 35</td>
<td>equipment list</td>
<td>List of capital equipment owned by the business</td>
</tr>
<tr>
<td>CI 36</td>
<td>SMP</td>
<td>The Safety Management Plan. This is a container for numerous Policy and Procedure documents which describe how to establish and maintain safe operating conditions. It is a legislative requirement that businesses maintain such a plan.</td>
</tr>
<tr>
<td>CI 37</td>
<td>SWI</td>
<td>A Safe Working Instruction is a specific directive issued by the Business to its employees, detailing some particular safety issue. They are often procedural, but are tailored to very specific tasks, scenarios or tools.</td>
</tr>
<tr>
<td>CI 38</td>
<td>OHS reports</td>
<td>A regular compilation of safety incidents, accidents and/or lost-time. It is a regulatory obligation that such records be maintained, and regular reports (typically monthly) be issued to employees. They are usually accompanied by safety procedure reminder excerpts, usually taken from the SMP.</td>
</tr>
<tr>
<td>CI 39</td>
<td>Company Newsletter</td>
<td>An informal regular (monthly) letter sent to all employees. It is used to maintain morale, to boost teamwork and to make announcements.</td>
</tr>
<tr>
<td>CI 40</td>
<td>Advisories</td>
<td>More formal and irregular letters sent to employees on a per-need basis. These typically advise of regulatory, environmental or procedural changes. The Business itself sometimes receives advisories itself from external agencies and instrumentalities, and is obliged to pass on such information to its employees.</td>
</tr>
<tr>
<td>CI 41</td>
<td>Employee Experience Survey</td>
<td>A short term device – used to elicit the current state of employee experience during the establishment of the experience-tracking process.</td>
</tr>
<tr>
<td>CI 42</td>
<td>Employee Induction Package</td>
<td>A set of instructional, educational and directive documents setting out necessary information and procedures for employees to follow and adhere to.</td>
</tr>
<tr>
<td>CI 43</td>
<td>GoogleMaps®</td>
<td>An online mapping tool able to estimate distance and travel time between address points</td>
</tr>
</tbody>
</table>

Table 5.5: Identified Business Documents

### 5.4 AT Analysis Phase

Having established a rough identification of positions, roles and candidate instruments, it is necessary to begin the AT analysis, wherein the extant business processes are described in AT terms. For convenience, the revised AT Taxonomy, as given in Table 4.8 is repeated here as Table 5.6 to assist the reader.

At the highest level, the business’ driving agenda (which informs all subordinate doings and thus provides the ultimate consistency check) is to provide competent specialist labour operatives in a timely manner, and match the needs of the client’s job. The activity level remains obscure and difficult to identify at this stage. The next
step in the ATSA method therefore is the identification, from Business processes, of GDAs.

<table>
<thead>
<tr>
<th>Doing</th>
<th>Facilitator</th>
<th>Driver</th>
<th>Product</th>
<th>Protagonist</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Activity Network</td>
<td>System</td>
<td>Agenda</td>
<td>Process</td>
</tr>
<tr>
<td>3</td>
<td>Activity</td>
<td>Tool</td>
<td>Motive</td>
<td>Outcome</td>
</tr>
<tr>
<td>2</td>
<td>Action</td>
<td>Screen</td>
<td>Goal</td>
<td>Transaction</td>
</tr>
<tr>
<td>1</td>
<td>Operation</td>
<td>Switch</td>
<td>Condition</td>
<td>Change</td>
</tr>
</tbody>
</table>

Table 5.6: Revised AT Taxonomy

### 5.4.1 Identify Goal Driven Actions (GDAs)

Actions are conducted by Position holders acting in a Role. Each requiring Instruments performing work (under conditions & constraints) to some purpose, producing Instruments(s) for others to use.

The identification of goal driven actions is a means of describing the transaction of candidate instruments between roles. As described above in section 4.24.2.1, initially identified Actions are rather short-lived. They serve as convenient loci for grouping the transactions of candidate Instruments between roles. The analyst must not become wedded to the early sets of actions as they’ll become divided into SINs, and later conflated into Activities. The AT precept that the activity is the base unit of analysis must be kept in mind at all times, even though the activity may not be easily identified before several steps of analysis.

It is entirely permissible for each action to have multiple Instruments entering and leaving; however there should ideally be only one single chief protagonist (actor) which will correspond closely to a position in a particular role. The analyst should feel free to reconfigure and rearrange the network of actions iteratively, under consultation with the stakeholders, until all parties reach sufficient agreement that a realistic representation of the group’s process has been achieved.

Each Action must be accompanied by its goal which, as described above in section 4.22.1.4, lists the actions from which Instruments are received and those to whom they are sent, together with all temporal and deontic constraints. It list the required components (candidate instruments incoming from other actions), the work that is done to, upon or with them, and the resulting candidate instruments. Any constraints and conditions must be faithfully recorded, especially those pertaining to the timing or sequencing of doings (temporal) or to do with obligation, permission or authorisation (deontic). These goal statements must be as clear and unambiguous as possible, as they are a kind of genetic information which will carry across to more refined interpretations of the network. As described in section 4.22.1.4, The general form of a goal statement should ultimately become: take instrument($X$) from role($A$), perform transform($T$) upon it under conditions($C$), and pass the resulting
instrument(X') to role(B). As further set out in section 4.22.1.4, these constraints should begin to indicate timing, sequence, ordinality, cardinality and obligation.

The listing of business procedures documented at the time of analysis is recorded in Appendix B. From these business procedures a number of initial goal Directed Actions were identified and selected for further analysis under the ATSA Method. As might be expected, the hierarchic location together with the degree of refinement and granularity of the descriptions as they appeared in the original Business process documentation, were inconsistent. Therefore this initial listing of GDAs given below in Appendix E already shows some rephrasing, omissions, contractions and expansions based upon direct explanatory elicitation from Business managers and staff. It is important to note that, as set out in section 4.22.1.5; these may be further promoted, demoted, split, joined or even omitted as the true extent of doings at the action layer becomes clearer. The analyst’s aim must be to identify Action layer Transactional events. During the re-design phase of course, any amount of variance is permitted so long as goals, constraints and conditions (eventually to be expressed under more formal terms as temporal rules and preferably deontic Divisions of Labour (DivLABs)) remain consistent with the group’s agenda and its motives.

The GDAs, presented in Appendix E, represent a first pass at identifying Action layer transactional events. GDAs were grouped under an alphabetical code where the letters A through F indicated likely grouping by business processes. The driving business purpose for these was recorded against each process header. It is important to note that these were not yet Activities, nor were the business purposes genuine AT activity motives, but the information was useful to later analysis, and so was retained here.

As may be seen even from the listing of selected GDAs presented in Appendix E, despite the business’ efforts to record and codify their processes, there remained redundancies and duplications. For example, during actions 26 to 28 in order for the EmpA to consult with an employee regarding details of proposed job site conditions, they may need to refer to the Job Booking Sheet or the more detailed Job Scope Report (which may not be completed at that stage).

Some actions drawn from the business process documentation, such as actions 7, 10 and 14, simply flag the commencement of further steps and/or the initiation of a fresh record or file entry. These were likely to be subsumed during further analysis as, effectively, non-actions. Likewise, action 2 was likely to be subsumed into action 1, and indeed, become a precursor sub-action. Action numbers for any of these possible changes however were not reassigned at this stage.

Whilst inadequate or incomplete process engineering produced some issues, some other matters were the result of, at the time, irreconcilable internal power structures. For example, the need to invest ultimate financial authority in the MD-Admin (acting as CFO and PM) and the need to distribute repetitive financial functions to low-authority roles such as the EmpA could result in clumsy workflows. During action 27 when employees were consulted on pay and allowance matters prior to being assigned a job, the EmpA was not empowered to re-open pay scale enquiries or negotiations with the client in the event of a query or dispute.
Since the business was still small, with very few office staff each filling multiple Roles and sitting in a common office space, such issues were dealt with by the simple expedient of face-to-face consultation (especially as each staff member was privy to the doings of all others). A well engineered process however should allow for smooth resolution of such not-uncommon exception cases even if the business should expand greatly and physically separate staff members and roles. At the time of analysis however, the business had not yet achieved sufficient organisational maturity to allow for a delegation of financial decision-making authority.

5.4.2 Decompose Actions to Single Instrument Nodes (SINs)

Each of the GDAs identified in the previous section were then decomposed into Single Instrument Node (SINs), each with only a single candidate instrument either entering or leaving. It was important that each SIN inherited the context and constraints of its parent action.

Of course, as analysis and design proceeded in later stages of the ATSA method, Instruments were refined until they represented computer system data and state entities. The SINs presented here are, at best, formative, representing largely unreformed doings and transacting with what are immature and essentially ill-defined candidate instruments.

As predicted in section 4.24.2.2 (above), there was a powerful temptation for the analyst to indulge in re-design at this stage. Detailed deconstruction of GDAs into SINs revealed duplications, inefficiencies and ambiguities which strongly provoked the consideration of possible changes. The analyst was obliged to avoid making changes. It was very important however that issues and ideas for future redesign were recorded as they arose.

The SINs extracted from the GDAs described in the previous section are shown in Appendix F below. The numbering of SINs follows that of their parent GDA (from Appendix E), assigning roughly sequential numbers to the component single instrument exchanged observed and noted by the analyst.

The table also identifies candidate Instruments with numbers from Table 5.5 above. It must be kept in mind that these candidate Instruments are drawn from extant business documentation and are very likely to be radically altered and expanded during the re-design phase when they are replaced with a larger number of real Instruments; these being, by prior definition, data objects transacted between stations and the system, across the interface.

Each SIN is given a brief description and some notes about its goal or the linkages it has to other doings. Side notes and comments are recorded against many SINs indicating issues encountered by the analyst, together with possible ideas for the re-design phase. The role most closely associated with the SIN is also identified.

5.4.2.1 Some Notes on the Decomposition to SINs

Upon more detailed analysis several doings became apparent which had not been fully described in the extant business process documentation. Some component doings of GDAs do not seem to give rise to any direct transactions of instruments
between roles. Unless the action contains vital constraint data which is not recorded elsewhere, these may not give rise to SINs.

It should be kept in mind that Instruments discussed in the analysis phase are only candidate instruments. Under re-design any number of changes is likely, especially the separation of candidate instruments into more object-like data entities. The final list of instruments after re-design is likely to be significantly larger, if only because transactions are being ‘bent’ through the system space, interacting with more, and more specific object-like data entities.

Here follows a number of notes taken during the decomposition to SINs, typical of the considerations and ideas the ATSA method anticipated could arise during analysis:

Note that some SINs, such as number 6.1, infer the need for multiple new SINs and doings not explicit in the extant business documentation. These should be addressed in the re-design phase as new and refined instruments and transformations are declared, and might be expected to produce multiple new doings.

Note that in SINs such as 9.2, the analyst has appended italicised notes towards future investigation and suggestive of considerations in the re-design phase yet to come.

Note that SIN 12.1 seems unnecessary or is out of sequence. Perhaps it should appear as a “checklist” type doing, perhaps under the “New Client” processes.

Note that SINs 13.1 and 13.2 do not appear in the business process documentation, but emerged during elicitation with the business. It was clear that some doing was required to initiate a job prior to its being confirmed (from GDA 12 as shown in Appendix E) Properly speaking, this clarification did not constitute a redesign, as it resulted merely from further clarification with the stakeholder, as permitted and indeed recommended under the ATSA method.

Note that in SIN 13.3, an unresolved temporal (sequencing) issue was noted; to be considered in redesign. This indicates that some SIN’s may appear (in the stakeholders’ organisational context) to be clustered in certain (as yet underdetermined) activities, but might ultimately be more efficiently conflated elsewhere.

Note that SIN 14.1 is another perfunctory doing, seemingly not worth recording, though there may be a need in the final system for some doing that initiates the recording of a job. Note further, that 14.1 now seems out of sequence, and perhaps should occur after 13.1 or 13.2 when a new job seems possible. Under the extant documented manual processes it was considered wasteful to manually start a new paper record before the status of a job became more certain. In a computer-facilitated system however, it may be just as easy to use the screen to hold temporary ‘notes’ whilst eliciting or receiving details of possible jobs; which can then either be simply deleted (if the job is lost) or then converted (elevated or promoted) to the status of genuine jobs. Such a mechanism may save double entry of data. This suggests the existence (under redesign) some kind of status flag instrument which indicates if a
job is active or not. Either the document changes its status or some internal transaction converts it to a new type of instrument; say from job-note to job-sheet.

Note that action 15 was more fully broken down in the stakeholder’s documentation than many other tasks.

Note that SIN 19.3 is unclear in the documentation, but was observed first hand by the analyst whilst observing the business in operation. The CSA had the authority to re-allocate employees to maximise the ability to service as many jobs as possible. Such reallocation would impact on multiple candidate Instruments, so this SIN technically breaches the definition of a SIN. This doing however represents an exceptional case which would be sufficiently rare to not required implementation into the system as an activity or doing per se, just so long as any editing of prior decisions is possible. This last scenario implies the usefulness of some kind of consistency checker to ensure all implicit effects of a change are actioned correctly and no employee, client or job is left in an untenable position.

Note that SIN 22.1 may be subsumed into earlier experience-based selection doings

5.4.3 Conflate Activities

The processing of GDAs to SINs and thence to activities; is the very heart of the ATSA method. These are the steps for converting or expressing a stakeholder Group’s doings in AT centric terms. This was the breakthrough solution to Leon’tev’s labyrinth (section 4.22.1.5). The goal of these steps is to render what is traditionally expressed in role or duty-centric terms, into doing-centric terms.

As described in section: 4.24.2.3 (above), the analyst must firstly sort SIN’s according to their protagonist (actor). These are still associated with positions, and probably still closely associated with Roles. It may be that the duty-oriented role descriptions no correlate closely represent logical do-ers, so the analyst must be prepared to redefine roles as necessary, so long as it is the same individual user (position) performing as subject of the conflated activity.

The analyst strives to sort SINs by their subject, and then sort again by those which could possibly exist under a single activity according to their object (motive). This is an imprecise, value-laden exercise on the part of the analyst.

Though prior to the final description of the ATSA method in chapter 4, the separation of analysis and re-design as distinct phases was not always clearly articulated; the notion that SINs might need to be re-expressed, refined or re-organised when conflating activities was clear. Section 4.22.1.5 noted the suggestion of possible activities from extant positions and roles (and by inference, their duty statements), and that the analyst might strive to enhance acceptance and learnability of the final system by following these original groupings. However, that early section acknowledged the analyst’s freedom to radically depart from the extant model as necessary.

There were holes in the recorded business processes, as might be expected from ad hoc documentation. During the conflation of activities, as suggested in the design and discussion of the ATSA method in chapter 4, the more methodological
mechanism of recording group doings tends to ‘self-correct’ and prompt an analyst to elicit further details as needed. For example, in this case, it became apparent that there was no initial record of how a New Client Application Form is initiated, nor by whom. It fell to the analyst to seek (if not suggest) clarification from the stakeholders.

It is important for the reader to realise that it is not necessary for activities to slavishly conform to the business processes as described by stakeholder documentation and opinion. Likewise, changes made to the organisation of SINs when conflating activities do not constitute redesign but are simply the expression of the stakeholder group’s doings in AT terms. The result is an AT-oriented Business Process Model (BPM) rather than the somewhat arbitrary and duty-oriented BPM the stakeholders themselves might otherwise envisage.

Each node now becomes a fully formed activity describable according to its Engström matrix elements. Recalling the Engström matrix diagram as shown in section 2.25.3’s Figure 2.15 (reproduced for convenience here as Figure 5.1) there are seven principle nodes to be detailed. Table 5.7 indicates a model activity shell, with which each of the conflated activities were summarised.

![Figure 5.1: AT matrix (from Engström (1987))]()

<table>
<thead>
<tr>
<th>ACTIVITY No.</th>
<th>Activity Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBJECT</td>
<td>the Role which ‘does’ this Activity</td>
</tr>
<tr>
<td>OBJECT</td>
<td>what the Activity wants to achieve (motive)</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>other Roles involved in this activity (recipients &amp; providers)</td>
</tr>
<tr>
<td>DivLAB</td>
<td>resources needed from others (often an Instrument)</td>
</tr>
<tr>
<td>RULES</td>
<td>constraints, conditions (things the Activity must satisfy)</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>actual result ~ effects other Activities (often an Instrument)</td>
</tr>
<tr>
<td>TOOL</td>
<td>psychological &amp; physical facilitators (including Instruments)</td>
</tr>
</tbody>
</table>

Table 5.7: Model Activity Shell

The activities conflated in this section of the ATSA method were each set out using a tabular pro forma based on the model activity shell given in Table 5.7 above. This provides a simple mechanism for record keeping. The ATSA Method itself does not currently prescribe or enforce any particular mechanisms of this kind, remaining as it
does deliberately agnostic towards all extant SAD, requirements Engineering (RE) and Software Engineering (SE) methods and techniques.

According to the ATSA Method’s design, there is a clear correlation between Role and subject, which was also reflected in the community node (listing other Roles with which the activity interacts.) As expected, the object (motive), DivLAB and rules required some interpretative re-expression on the part of the Analyst, making use of further iterative consultation with the stakeholder(s).

A simple spreadsheet was employed to assist in the conflation of activities. It served as a scratch board on which SINs could be gathered and sorted by role. They were then further sorted manually according to the analyst’s understanding of the Goals and constraints. The resultant worksheet is presented completeness in Appendix G. Here thirty nine activities, clustered into eight broad processes were identified.

In accordance with the earlier discussion on the selection of a suitable body of data for the exploration of the ATSA method (section 5.3.2.1), Thirty five activities (clustered in six broad processes) were chosen to be written up in tabular ‘shell’ format. These are presented in Appendix H.

Whilst the activity shells or some similar representation (perhaps a database entry) for each activity is required, representation of the activity network as a directed graph is not a formal requirement of the ATSA method as detailed in chapter 4. Whilst such a diagram served well in the trial case thought experiment (chapter 4) it could quickly become impractically large in real world cases. For the purposes of assisting the analyst to present and discuss the ATSA AT-centric BPM to the stakeholder however, a graphical layout may offer some benefits when used in conjunction with the tabular activity shells.

Some future suite of semi-automated tool(s) for elicitation analysis and re-design under the ATSA method might be imagined to offer a graphical interface with scrolling and zooming. Such a tool could ease the creation and display of large network diagrams perhaps under a multi-touch gestural paradigm. Further consideration of the possibilities offered by these new interface technologies was conducted in chapter 7 (below).

As previously observed, the adjacency matrix of the activity network constitutes a Combined Activity Table (CAT). Mathematically the CAT and the network are equivalent and each may be produced readily from the other. Whilst a network diagram proved impractical in this case, Appendix I presents the CAT for the test case. It is worth noting that whilst the CAT lists the thirty five activities, a number of transactions (such as the storage or retrieval of employee files in the Employee Folder) were most easily represented by presenting the entity ‘Folders’ on the CAT. Likewise ‘Boards’, ‘Cache’, the external groups ‘Employees’ and ‘Clients’ and the external 3rd party softwares ‘MYOB®’, ‘FAID® and ‘online banking’ were shown as entities capable of receiving or sending candidate instruments. This may represent the need to expand slightly upon the ATSA method, as the presence of such non-activity entities could be good indicators of the potential to bend transactions through the yet-to-be-described system.
5.5 Re-Design Phase

The stepwise overview of the ATSA method as described in section 4.24 and briefly recapped in Table 5.1 (above), the re-design phase constitutes six elements: Running Node Reduction Heuristics (NRHs), bending instrument paths to system, converting rules, the converting of goal statements to Pseudocode, assembling the instrument list and finally the identification of switches and combining them into screens etc.

The ATSA method was intended to be agnostic of extant SAD and Human Computer Interaction (HCI) methods, tools, techniques and methodologies, in anticipation that the resulting method might be adaptable to, or at least not irretrievably incompatible with, a broad range of existing approaches. As a result, the ATSA method does not rigidly follow the classic stepwise structured software development lifecycle as a matter of course. Therefore, it does not prescribed that each of its six steps must be run discretely and/or sequentially across the entire subject case.

Whilst ATSA has no prescribed alignment to any particular software development lifecycle, it has tended towards an iterative waterfall with non-linear evolutionary aspects. It is envisaged that, as complex systems tend to have clusters of almost separable doings (ultimately expressed as activities & actions) within the greater whole, several of the ATSA steps might perhaps be run across clusters of doings. The final steps of assembling the instrument list and switch identification and combination however are most likely to occur at the end of the phase, but even then, elements and ideas towards these final steps will tend to have accreted during the preceding steps.

<table>
<thead>
<tr>
<th></th>
<th>Phase 3 Meaning after redesign</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTIVITY</td>
<td>System Boundary Nodes: transact Instruments, contain interfaces.</td>
</tr>
<tr>
<td>SUBJECT</td>
<td>The User at their Station</td>
</tr>
<tr>
<td>TOOL</td>
<td>Switches, Screens, Screen-Sets &amp; Stations</td>
</tr>
<tr>
<td>OBJECT</td>
<td>Motive statement that encapsulates SET of Goals</td>
</tr>
<tr>
<td>RULES</td>
<td>Mostly converted into DivLAB requirements</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>The System is now the entity with whom all Instruments are transacted.</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Activity inputs</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Activity outputs</td>
</tr>
<tr>
<td>ROLE</td>
<td>Replaced by Sets of Subjects, conflated in Stations</td>
</tr>
<tr>
<td>POSITION</td>
<td>The user. Subject of one or more Activities, via bespoke Station</td>
</tr>
<tr>
<td>INSTRUMENT</td>
<td>Data objects exchanged between Stations &amp; System, through Interface(s)</td>
</tr>
<tr>
<td>SYSTEM</td>
<td>DivLAB + Outcome : list data(s) in &amp; out Tool + Object : describe transforms and interfaces</td>
</tr>
</tbody>
</table>

Table 5.8: Meanings of Method Terms after Redesign
As discussed above in section 4.23, the meaning of the Engström nodes and other method terms changes between the three phases of the ATSA method. Table 5.8 below (an extract from Table 4.10) presents the meanings of method terms after the redesign phase. It is crucial to note that true instruments are clearly limited to those data objects which pass between stations and the system through the interface and that true activities are those which make use of such instruments and are described in terms of their instrument requirements.

The goal of the re-design phase therefore, aside from improving the group process experience for the client stakeholders of course, is to express the system in terms consistent with those given in Table 5.8.

Phase three of the ATSA method presents the analyst (now perhaps best described as designer) with a somewhat creative task. To imagine how the given BPM might be improved by describing the requirements of a system through which instruments would be rerouted, could well be deemed a subjective activity. It is quite reasonable to assume that more than one design might be produced. As stated earlier however, the ATSA method makes no claim at this time to produce optimal system designs. The goal of this investigation is simply to investigate if an AT based SAD method is at all feasible. The author, therefore, makes no claim as to the elegance or efficiency of the redesign solutions proposed in this section and craves the indulgence of those readers with greater experience of system design and or with varying sensibilities.

5.5.1 Node Reduction Heuristics (NRHs) and Bending Instrument Paths through the System

As described in section 4.24.3.1, the designer should identify pipes, which are those transactions between activities which have a common Subject; and joints which are Activities through which an instrument passes unchanged. Locations on the activity network (in effect, a BPM as described above) where pipes and joints coincide are strong candidates for collapse and or redesign. The designer must pay particular attention to goal and motive constraints to ensure that any modification of transactions does not compromise the group process’ agenda.

Appendix J shows a modified CAT, which presents a tabular listing of instrument transactions by activity. New columns have been added to record pipes and nodes. The designer has appended arrows to indicate the activity to activity transactions (something which might have been more clear in a diagrammatic activity network diagram), and has recorded if each transaction constitutes a pipe or not. Note that there are very few pipes.

The modified CAT in Appendix J also indicates possible joints, by circling those Activities through which any instrument is both received and sent (ie: neither created, nor consumed). Genuine joints have been identified in the table. Note that there are very few joints, as the Instruments involved are almost always modified or impacted in some way.

After applying the NRHs, the Designer has recorded on the modified CAT, in columns set aside for that purpose, if a pipe or joint is collapsible.
Chapter 5 – Method Evaluation

Though the stepwise description of the ATSA method in chapter 4 described the NRHs and the bending of instrument paths through the system in separate sections, in practice it served better to run them paired. The heuristics served to assist the designer in identifying areas to simplify, which then prompted the redirection of Instrument transactions. The designer retained a second copy of the modified CAT (presented in the attached CD-ROM spreadsheet), on which further Redesign Thoughts were recorded, based upon the NRHs and on possibilities which arose whilst node reduction was considered.

From section 4.24.3.2, as the location and boundary of the potential system becomes apparent, the designer should actively seek to reroute (bend) the instrument transactions through it. The notion is to consider that instruments reside inside the system. Activities around the system boundary should exchange instruments with the System wherever possible, rather than directly with each other. As described in chapter 4, the designer is free to split, joined or create activities as deemed necessary to facilitate this. Whenever this occurs however, their component SINs must be considered, and all effort made to ensure the motives and goals are not compromised. The reader is reminded that there was no ‘system’ as yet. Its nature would emerge as the designer was able to assign functional details to it.

The primary function of the second modified ‘redesign thoughts’ CAT was to note which doings might best be automated. It also acted as a worksheet or scratchboard where the designer could consider how post redesign Instruments might be connected through the system.

Following the reduction of nodes and the realignment of instrument transactions, some activities, seen to no longer conduct any transactions across the system boundary must be deemed peripheral activities and cease to be directly any part of the system. Details of peripheral activities should be retained however, to provide useful context and to facilitate any future revision or redesign.

In accord with the underlying concepts of the ATSA method, transactions were identified which could (if activities were joined, merged, split or otherwise modified) have both termini within the system, thus in effect automating them. Such doings would then become internal to the system, if effect system activities, for which the system itself served as subject (protagonist).

The designer also sought to identify and exclude transactions which are unlikely to ever have either terminus crossing the system boundary as the objects transacted can, by definition, not be true instruments.

Only those instruments which would reside in system could ultimately be deemed true instruments, as distinct from the candidate instruments discussed up until this point. These might be thought of as closely corresponding to data objects and variables in the final System build, but the ATSA method deliberately tries to avoid such labels in order to retain paradigmatic neutrality. Under variation, it is imagined that an Object Oriented (O-O) approach could apply at this point. Instrument transactions and their constraints could also inform the identification of potential object classes and transactions. It was deemed beyond the scope of this current research to investigate this option further.
5.5.2 Convert Temporal Rules into Deontic DivLABs and Convert Goal Statements to Pseudocode

Under the ATSA method, true instruments may be imagined as inhabiting a generic System Data Repository (SDR). The contents of this SDR, when complete, will comprise the Instrument list. Note that the assembly of the instrument list is not accorded a separate step in the method description; rather, true instruments accrue and are refined and defined throughout and across all the steps of the redesign phase. As transactions are bent across the system boundary (invoking the system as an Actor), the designer should also seek to more accurately define the true instruments into recognisably computable components.

As set out in section 4.24.3.3, following any modifications from the preceding stages, Activities should retain a complete suite of constraints, conflated from the inherited characteristics of their component SINs. The rules of these SINs consist largely of temporal (sequential) constraints of the general form ‘must do during state S’. The designer strives to rewrite as many of these rules as possible into the more deontic flavoured DivLAB constraints. These are essentially preconditions, preventing the execution of a doing until required conditions are satisfied. This places an obligation upon the provider of an instrument (the system itself, in every case) to maintain and monitor their state. The degree to which rules are converted into DivLABs is considered an indication of the reliability of the final system, as DivLABs may be built directly into the functionality of the system.

In this process, SINs may be profoundly modified. Afterwards, many of SINs occurring within activities will take the form of receivers. New SINs may thus be created, and many will be rewritten. Care was taken to ensure accurate inheritance (under conversion or translation, as required) of SIN characteristics, conditions and constraints. It was also preferable to express, or convey, the goal statements into simple generic pseudocode (structured English) wherever possible. This encapsulated something akin to the functional requirements for each Instrument.

In the case study, it was found that during the collection of redesign thoughts and recording them onto a modified CAT (described in the previous section), it was convenient to begin describing instruments and their constraints and conditions simultaneously.

The second copy of the modified CAT, on which further Redesign Thoughts were recorded, was of course listed by activity, but as it became apparent that the behaviour of instruments could be readily deduced, an Instrument ordered listing was used as well.

In declaring where (by activity, subject and in relation to which other instruments) each instrument is declared (created) and used (consumed, edited, read, modified etc), the instrument list essentially described the pre-requisite conditions and other constraints. In short, the assemblage of the instrument list during these preceding and quasi-concurrent steps, contained satisfaction of the conversion of rules to DivLAB preconditions, and the expression of goals into structured English.
As an example of how goals and motives (objects) may be expressed in the system, consider activity 15, “Track Start Date” (recorded in the CD-ROM spread sheet). The object for this activity is:

“Activate” the job for tracking. Ultimate motivation is to book a job for the business to service by providing skilled specialist employee(s).

To reflect this object in the instrument list, a specific instrument ‘jobs’ was created with the following properties:

‘Jobs’ was declared to be of collection type and contained a component called ‘status’, which itself was of the bespoke datatype JOB-STATUS; declared to be an ENUM (enumerated value) from the value set: “Pending”, “Current” or “Ended”. Structured-English notes recorded that:

\[ \text{if } (\text{today})<\text{(jobs(i).start)} \text{ then "pending" else } (\text{jobs(i).end}) \text{ "current" else "ended"} \]

As an example of how rules may be converted to DivLAB statements in the population of the instrument list, consider Activity 6, Evaluate Potential Employee (recorded in the CD-ROM spread sheet). The recorded rules for this activity were that:

Tickets, medical and PPE are pre-requisites for selection.
Referee reports may be over-ruled.
CSA may consult with potential employee, senior employees with personal knowledge and/or industry contacts in lieu of referral reports.
Final suitability decision and initial status and condition decided by CSA.
Default initial status is AMBER.
Less experienced new employee may require supervision.

To reflect these rules in the instrument list, the following mechanisms were inserted:

Whenever a potential employees qualification tickets are within 30 days of expiration, an automated Instrument ‘qual_alert’ is presented to the EmpA. Once the current date exceeds the expiration date of a qualification, the now unqualified employee’s ID is automatically removed from the instrument which is a collection of employees holding that particular qualification (and so has become unavailable for selection).

A similar automated mechanism lists an employee’s medical status as invalid if their medical certification has expired.

Whilst referee details and reports are recorded, only the CSA has access to entering the final decision to employ or reject.

The default value of each new employee’s status is set to AMBER.

There are instruments which record the experience accruable in a number of fields for each job. The instruments employed in employee selection allow for any number of decision filters to be set, including specification
of a specified amount of experience. This facilitates any decision to assign specific job roles within a placement, such as that of supervisor.

### 5.5.3 Instrument List

As described in the preceding sections, the assemblage of the instrument list served as the culmination of the four preceding steps; applications of the NRHs, the bending of Instrument paths through the system, conversion of rules into preconditional DivLABs and the conversion of goal statements to pseudocode.

The instrument list is where the redesign thoughts were instantiated and given form. The declaration of preconditions and goals for instrument declaration and use (in the context of actions, subjects and other instruments) prompted the creation and refinement of other instruments, as necessary.

Adopting, for simplicity, convenience and broadest comprehension, a general taxonomy found in many traditional structured system design techniques, instruments in this case study were identified with data entity entries in a table much like a data dictionary. Each was allocated a short alphanumeric name which reflected its core function in terms of the redesigned activity network. Table 5.9 below, shows the field headers used to describe Instruments.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Description</th>
<th>BINDINGS</th>
<th>WHERE (Activity) (Role)</th>
<th>Coding Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Var vs Const</td>
<td>Local vs Global</td>
<td>Static vs Dynamic</td>
<td>Scalar, Collection, Datatype</td>
</tr>
</tbody>
</table>

**Table 5.9: Field Descriptors used in the Instrument List**

In Table 5.9 above, the instrument field contains the alphanumeric name assigned to each instrument. The description field contains a very brief comment to provide context.

The binding fields record if an instrument data entity is of type INT (integer), FLOAT (fractional value), VAR (variable), STRING (text block), BOOL (Boolean), ENUM (enumerated type), BITMAP (graphic file), COLLECTION (deliberately non-specific multiple entry entity which might be an array, list etc), STRUCT (structure of component data elements), FORM, REPORT etc. they also record if an instrument data entity would contain a variable or constant value, would exist globally throughout the system or would only exist locally to a given activity (analogous to ‘function’ or ‘module’), would be static or dynamic (invariant between activities), or would consist of a single value or an array of multiple values.

For each instrument, the designer records where it is initially declared and where it is used (called, read, deployed etc). The ‘locations’ for declaration and use are given in AT terms, by activity. For each location, the designer indicates the activity (for ease of differentiation, this section arbitrarily indicated by the use of square brackets [~]), the users’ role (indicated with braces {~}) and as necessary, any references to other Instruments (indicated with parentheses (~)).
Finally, the Coding Notes field presented an opportunity for each Instrument to be described in a process-oriented manner, using somewhat structured English suggestive of pseudocode. For example, in Table 5.10 below, sample entries for some interacting Instruments are given.

The instrument ‘today’ is an integer variable used globally; with a dynamic value that is automatically set by the system clock. (It is deemed a reasonable assumption that any modern computer system would possess a system clock timekeeping feature). Whilst ‘today’ is used in many places, for simplicity, Table 5.10 only records its use in an automated doing (i.e. one conducted entirely within, and by, the system itself), involving a Instrument called ‘acc_lett’, and again in a component of the collection instrument ‘emps’, called ‘med_status’.

The instrument ‘emps_med_status’ itself is a BOOL (Boolean; true/false) global static scalar variable, initially set by the {HumA} role in activity [7] when a new employee is established in the system. It indicates if a given employee holds current medical certification, which is an important criterion for the selection of employees for any given job. The Coding Notes entry indicates that the system will automatically set the value of this Boolean variable to false if the ‘today’ value is greater than the ‘emps().med_exp_date’ value, which is a static integer variable, set by the {HumA} during activity [7], and which records the expiry date of the employees current medical certification. This simple expression ensures that an employee with an expired medical status is marked as such and operates in accord with the goals and motives of the SIN’s, actions and activities elicited from the group process.

Some instruments (in this sample, ‘apps’ and ‘emps’) are classified as collections. This term indicates that the Instrument contains a number of component values. Whilst typically such an entity might be labelled as an array, list or the like, the ATSA method strives to remain agnostic of implementation technique. The decision as to how best to declare the Instrument in code is left up to the coder, and not specified in these high level specifications. For each collection, a count instrument (typically ‘n_collection_name’) is declared, being simply the size of the collection. Regardless of implementation, the size of a collection will be of use for restricting access only to valid components.

The Boolean components of the collection instrument ‘apps’; ‘apps(i).decided’ and ‘apps(i).accepted’ (being the true/false values of the components ‘decided’ and ‘accepted’ for the ith application in the collection ‘apps’), record if a decision has been made to employ them or not, and if that decision was positive. They are set when the Chief Safety Auditor {CSA} makes a determination to accept or reject an applicant for employment.

Many collections (such as ‘apps’ and ‘emps’ in this sample) are collections of structured data. In such cases, the collection is of a type named for a bespoke STRUCT datatype (declared to define the collection’s components). In this sample the datatypes ‘APP’ and ‘EMP’ are presented. The components of these are, strictly speaking, not typical of true instruments in and of themselves. It is notable that different components of a STRCT might be declared or used in different activities and by different roles. This is one way in which the analyst/designer might track and
record precisely how each instrument is interacted with under each transaction it passes through.

The instrument ‘acc_lett-tmp’ is a FORM document, controlled by the Document Control Officer {DCO} in one or more infrequent ‘maintenance’ activities not recorded in this current analysis. It is stored in the system, and retrieved when a decision to employ is entered (setting the value of ‘apps(i).accepted’ true). Although ‘acc_lett_tmp’ is assigned as a constant, being typically remains invariant once initialised by the {DCO}; strictly speaking FORMs of this type might vary under infrequent document revision activities which are outside the scope of this current analysis.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Description</th>
<th>BINDINGS</th>
<th>WHERE [Activity] (Role)</th>
<th>Coding Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>today</td>
<td>Current date</td>
<td>INT, Sys Var Global, Global</td>
<td>Dyn, Scalar [AUTO], [AUTO]</td>
<td>From the system clock</td>
</tr>
<tr>
<td>apps</td>
<td>Listing of employment applications</td>
<td>APP Var Global Static, Collection</td>
<td>[INIT]</td>
<td></td>
</tr>
<tr>
<td>n_apps</td>
<td>No of employment applications</td>
<td>INT Var Global Static, Scalar</td>
<td>[AUTO] =size(apps)</td>
<td></td>
</tr>
<tr>
<td>APP</td>
<td>Employment application record</td>
<td>STRUCT ... Global ... Datatype [INIT]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>decided</td>
<td>Decision made yes/no</td>
<td>BOOL Var Global Static, Scalar […] (CSA)</td>
<td></td>
<td>Default value = False</td>
</tr>
<tr>
<td>accept</td>
<td>Decision to employ yes/no</td>
<td>BOOL Var Global Static, Scalar […] (CSA)</td>
<td></td>
<td>Valid IFF apps(i).decided = True</td>
</tr>
<tr>
<td>acc_lett-tmp</td>
<td>Acceptance letter template</td>
<td>FORM Const Local Static, Scalar [MAINT] (DCO)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>acc_lett</td>
<td>Acceptance letter</td>
<td>REPORT Var Local Dyn, Scalar [AUTO]</td>
<td></td>
<td>Valid IFF apps(i).decided = True</td>
</tr>
<tr>
<td>emps</td>
<td>Listing of employees</td>
<td>EMP Var Global Static, Collection [INIT]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>n_emps</td>
<td>No of employee records</td>
<td>INT Var Global Static, Scalar [AUTO] =size(emps)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EMP</td>
<td>Employee record</td>
<td>STRUCT ... Global ... Datatype [INIT]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>med_status</td>
<td>BOOL Var Global Static, Scalar [7] (HumA) […]</td>
<td>False if (today) &gt;</td>
<td>emps(i).med_exp_date</td>
<td></td>
</tr>
<tr>
<td>med_exp_date</td>
<td>INT Var Global Static, Scalar [7] (HumA) […]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.10: Some Sample Instrument Records

The instrument ‘acc_lett’ is a single instance of an employment acceptance letter, auto generated as a mail-merged hardcopy report output when ‘apps(i).accepted’ is set to true. Generation of the local and dynamic ‘acc_lett’ requires the ‘acc_lett_tmp’ FORM, the successful applicant’s name and contact details (stored elsewhere in Instrument components; apps(i).fname, apps(i).lname and apps(i)addrs (not shown...
here)) and the current date from Instrument today. In a non-system human-only activity, this hardcopy output would be reviewed, signed and posted by an officer of the business. The specific mechanism to ensure the acceptance letter is only generated once (and not continuously after ‘apps(i).accepted’ becomes true) is an implementation-time decision for the coder.

The final instrument List for the system under redesign was drawn up as a simple spreadsheet, and is presented on a CD-ROM attached to original editions of this work submitted for assessment, and (it is anticipated) available on request for future readers.

5.5.4 Identify Switches and Combine them into Screens etc

The identification of switches and screens and their assemblage into a functional UI is a non-trivial process. The task is essentially an implementation issue for the HCI builder (much as final construction of the system itself falls on the coder’s implementation of the requirements specification), and ATSA makes no claim to provide a tool suite to guide such work. ATSA does claim, however, to provide sufficient information to facilitate such work, especially by neophyte designers who might otherwise want for a starting point.

No claim is made as to the optimality of any interface design suggested or informed by an ATSA-based analysis and system design, nor of the elegance of this author’s sample UI solutions (presented below), beyond the claim that ATSA, by identifying switches by parent activity, allows the HCI builder to cluster and group on-screen controls in a manner well suited to the users’ expectations and the nature of their doings, and further, describes data-like instruments in association with their use in a manner which may be suggestive of UI elements.

It was hoped, from the earliest stages of this research project, that an AT informed SAD would exhibit some indications of synthesis between system specification and UI specification. It is claimed that, if the analyst/designer (and/or system or UI designer/builders) follows the ATSA method, they should find that changes they may make in the system design are almost immediately suggestive of changes to the UI and visa versa.

For the purposes of demonstrating how the ATSA method can assist in the construction of a UI, a representative sample of data was drawn from the case study detailed in this chapter.

It was observed during the case study that junior staff, such as the Administrative Assistant, required broad but shallow access to many activities. Authority and decision making opportunities were severely limited for low rank positions, but they were still required to touch upon many doings, if only in a purely perfunctory and procedural capacity. Directorial staff could likewise be exposed to a broad range of activities. Whilst they retained the authority to delve into the details of any transaction, they tended to chose to remain aloof and simply oversight the doings of the business process, which resulted in a similarly shallow exposure. This research study did not concern itself with designing a Decision Support System (DSS) to assist senior managers by providing any specific form of dashboard, though such a project aught to be within the capabilities of the ATSA method.
For the purposes of investigating ATSA’s HCI value, it was decided to examine the position of Office Manager (OM) in closer detail as it was a middle management position with multiple, but specific authorised duties. At the time of analysis, this position performed eight identified roles (as detailed in the spreadsheet on the attached CD-ROM). For clarity the relevant excerpt is reproduced here as Table 5.11 (below). Two particular OM’s roles were chosen, being the Job Booking Administrator (JobA) and the Employment Placement Administrator (EmpA), because they had been examined in some depth in the preceding sections, and had some interesting interactions.

<table>
<thead>
<tr>
<th>(MO)</th>
<th>Marketing Officer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Supported by the AA, under direct supervision of the MDs</td>
</tr>
<tr>
<td></td>
<td>Liaises with potential Client Companies. Offering services following opportunities identified in Tender announcements, industry media, word-of-mouth etc.</td>
</tr>
<tr>
<td></td>
<td>Note: All external representations of Business services are co-ordinated with the (PRO) Public Relations officer and (SLO) Strategic Liaison Officer, and approved by an MD.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(CLO)</th>
<th>Client Liaison Officer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Supported by the AA</td>
</tr>
<tr>
<td></td>
<td>Liaises with Client Companies: checking the Scope of Work bookings; checking job Commencement and/or Completion; checking Customer satisfaction</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(JobA)</th>
<th>Job Booking Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>May also be performed by the MD-Opns, or the AA (under supervision)</td>
</tr>
<tr>
<td></td>
<td>Takes enquiries from Clients and books the details of requested jobs.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(EmpA)</th>
<th>Employee Placement Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>May also be performed by the MD-Opns, or the AA (under supervision)</td>
</tr>
<tr>
<td></td>
<td>Allocates employees to specific jobs, being careful to meet all operational and administrative requirements.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(DCO)</th>
<th>Document Control Officer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Maintains records of corporate files and documents, including Policy and Procedural documents, tracking currency, availability, completeness and version control. The DCO also maintains and manages copies of external information resources, such as industry relevant Standards, Legislation and Regulations.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(ICO)</th>
<th>Internal Communications Officer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NB: may also be conducted by the MD-Opns, or the AA</td>
</tr>
<tr>
<td></td>
<td>Co-ordinates internal distribution of information including: employee newsletter; regular OHS reports and advisory post outs; emergency medical contacts; onsite work papers; policy and procedural updates; logging corporate correspondence.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(OffA)</th>
<th>Office Supplies Administrator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Duties: This person tracks supplies of stationery and maintains stocks of necessary materials, inks and similar materials. All procurements must be approved by the Chief Finance Officer.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(OHS)</th>
<th>OH&amp;S Co-Ordinator</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Oversees OH&amp;S inspections and compliance within the corporation, most especially within the office environment. Chair meetings of the OHS committee and write corporate OHS reports. This person will hold, or be studying towards, an appropriate OH&amp;S certification.</td>
</tr>
<tr>
<td></td>
<td>Note: ‘OH&amp;S’ (Occupational Health and Safety) refers to New South Wales Legislative obligations upon all employers to maintain safe working environments.</td>
</tr>
</tbody>
</table>

Table 5.11 Roles currently assigned to the Office Manager

By selecting roles which were instantiated one individual and which had some interactivitivities within the group agenda, it was hoped that the relevance of the
ATSA method’s “5S” notion of system, station, screen-set, screen and switch could be examined.

Drawing from the more complete analysis and design information provided in the CD-ROM spread sheet, a table of activity shells for the JobA was extracted (Table 5.12) and another for JobA’s ‘real’ instruments (Table 5.13). Likewise, a table of activity shells for the EmpA (Table 5.14) was extracted, along with a table of the ‘real’ instruments used by the EmpA after re-design (Table 5.15). In both of these instances of role data, the reader is reminded that the activity shells predate redesign, but still serve to provide context for the use of instruments and the clustering of the necessary switches and screen widgets.

As given below in Table 5.12, the JobA serves as the subject for pre-re-design activities 12, 14, 15 and 16. As recorded in the redesign-thoughts page of the spread sheet file on the attached CD-ROM, activity 12 is where details of an upcoming job are elicited and recorded, whilst in activity 14 the job sheet is ‘activated’ and placed enters a manual sequence of notice board postings.

Under redesign, activities 15 and 16 became entirely automated and no longer required user attention as the status of each job (as Pending, Active or Ended) is set by the system according to the start and end dates and no longer needs to be manually tracked and indicated by the use of multiple notice boards.

<table>
<thead>
<tr>
<th>ACTY - 12</th>
<th>Complete Job Scope Report</th>
</tr>
</thead>
</table>
| SUBJECT   | JobA (Job Administrator) : [Current Position: Office Manager]  
Client (as necessary) |
| OBJECT    | To promptly elicit and record all details of upcoming Job, so it can be serviced to the highest standard. Ultimate motivation is to book a Job for the Business to service with its skilled specialist employee(s). |
| COMMUNITY | GenA (General Administrator) : [Current Position: AA]  
CFO (Chief Financial Officer) : [Current Position: MD-Admin]  
EmpA (Employee Placement Administrator) : [Current Position: OM]  
Client |
| DivLAB    | Receive (CI 18) Job Booking Sheet from GenA [10]  
Receive completed (CI 19) Job Scope Enquiry Form from Client.  
Receive elicited details from Client  
(possibly contact for more details and/or clarification) |
| RULES     | Promptly record (eliciting as required) as many pertinent details as possible.  
Advise CFO to initiate financial entry.  
Advise EmpA to initiate employee selection. |
| OUTCOME   | (CI 20) Job Scope Report: record necessary Job details.  
(CI 20) Job Scope Report attached to (CI 18) Job Booking Sheet  
(henceforth treated as one document)  
Updated (CI 18) Job Booking Sheet send to CFO [13] |
| TOOL      | (CI 18) Job Booking Sheet  
(CI 19) Job Scope Enquiry Form  
(CI 2) Client File  
(CI 20) Job Scope Report  
Letter, Fax, Telephone, eMail, Personal Conversation etc. |
### ACTY - 14  
**Create and post Job Sheet**

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>JobA (Job Administrator) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>&quot;Activate&quot; the Job for tracking. Ultimate motivation is to book a Job for the Business to service by providing skilled specialist employee(s).</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>CFO (Chief Financial Officer) : [Current Position: MD-Admin]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Receive (CI 28) Invoice Number From CFO [13]</td>
</tr>
<tr>
<td></td>
<td>Read Charge rate from (CI 2) Client File (stored in Client Folder).</td>
</tr>
<tr>
<td>RULES</td>
<td>Record Invoice Number onto Job Booking Sheet.</td>
</tr>
<tr>
<td></td>
<td>Record charge rates onto Job Booking Sheet.</td>
</tr>
<tr>
<td></td>
<td>Compile “half-page summary” Job Sheet from Job Booking Sheet.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>(CI 22) Job Sheet created with pay rate and (CI 29) Invoice number recorded, posted to Pending Jobs Board.</td>
</tr>
<tr>
<td></td>
<td>Advise EmpA [17] of (CI 22) Job Sheet</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 28) Invoice Number</td>
</tr>
<tr>
<td></td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td></td>
<td>(CI 2) Client File</td>
</tr>
<tr>
<td></td>
<td>(CI 22) Job Sheet</td>
</tr>
<tr>
<td></td>
<td>(CI 23) Pending Jobs Board</td>
</tr>
</tbody>
</table>

### ACTY - 15  
**Track Start Date**

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>JobA (Job Administrator) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Ensure Job Status is correct so Job can be serviced correctly and promptly. Ultimate motivation is to provide timely and accurate service to Client, to retain Client's future patronage.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>-</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Read (CI 22) Job Sheet from Pending Jobs Board</td>
</tr>
<tr>
<td></td>
<td>Read current date from calendar</td>
</tr>
<tr>
<td>RULES</td>
<td>Job Sheet shows accurate Start Date.</td>
</tr>
<tr>
<td></td>
<td>Current date ≥ Job end date (close as possible)</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>(CI 22) Job Sheet moved from Pending to Current Jobs Board.</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 22) Job Sheet (*) Start date</td>
</tr>
<tr>
<td></td>
<td>(CI 23) Pending Jobs Board</td>
</tr>
<tr>
<td></td>
<td>(CI 24) Current Jobs Board</td>
</tr>
</tbody>
</table>

(*) In effect this is a simple status-flag change on the Job Sheet (itself merely a representative token for the Job booking Sheet) – HOWEVER the Job Boards serve as important user interfaces, so the status-flag change MUST be accompanied by a visible change in where the Job Sheet (token) appears. The Job Boards are just Visibly Displayed Folders, containing Job Sheets according to their date status.
Chapter 5 – Method Evaluation

ACTY - 16  Track End Date

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>JobA (Job Administrator) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Ensure Job Status is correct so Job can be serviced correctly and promptly. Ultimate motivation is to provide timely and accurate service to Client, to retain Client's future patronage.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>CLO (Client Liaison Officer) : [Current Position: Office Manager]</td>
</tr>
</tbody>
</table>
| DivLAB | Read (CI 22) Job Sheet from Current Jobs Board  
Read current date from calendar |
| RULES | Job Sheet located on Current Jobs Board shows accurate End Date  
Current date ≥ Job end date (close as possible) requires a daily check. |
| OUTCOME | (CI 22) Job Sheet removed from (CI 24) Current Jobs Board  
Forwarded to CLO for post-job processing [31] |
| TOOL | (CI 22) Job Sheet (*) End date  
(CI 24) Current Jobs Board |

(*) In effect this is a simple status-flag change on the Job Sheet (itself merely a representative token for the Job booking Sheet) – HOWEVER the Job Boards serve as important user interfaces, so the status-flag change MUST be accompanied by a visible change in where the Job Sheet (token) appears. The Job Boards are essentially just Visibly Displayed Folders, containing Job Sheets according to their date status.

Table 5.12: Activity Shells for JobA

As indicated, the activity descriptions given in Table 5.12 predate the redesign process. A series of possible redesign thoughts were recorded in the attached spreadsheet and they informed the declaration of the ‘real’ data-like instruments also recorded on the spread sheet.

Whilst the activity shell records the motive for the JobA’s doings, it is the final list of real instruments which the JobA will deal with. The motive and the instrument transactions together inform a possible selection and assembly of UI elements.

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Description</th>
<th>WHERE [Activity] {Role}</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name</td>
<td>[12] (JobA)</td>
</tr>
<tr>
<td>description</td>
<td>Description</td>
<td>[12] (JobA)</td>
</tr>
<tr>
<td>descr</td>
<td>job description</td>
<td>[14] (JobA)</td>
</tr>
<tr>
<td>location</td>
<td>job Location</td>
<td>[14] (JobA)</td>
</tr>
<tr>
<td>map</td>
<td>job Map</td>
<td>[14] (JobA)</td>
</tr>
<tr>
<td>start</td>
<td>job Start Date</td>
<td>[14] (JobA)</td>
</tr>
<tr>
<td>end</td>
<td>job End Date</td>
<td>[14] (JobA)</td>
</tr>
<tr>
<td>status</td>
<td>current status of particular job, according to current date and scheduled start/finish dates</td>
<td>[AUTO]</td>
</tr>
<tr>
<td>selection</td>
<td>final selection determined?</td>
<td>[..] (JobA)</td>
</tr>
<tr>
<td>qual</td>
<td>Qual per person needed</td>
<td>[12] (JobA)</td>
</tr>
</tbody>
</table>

Table 5.13: JobA Instruments

Figure 5.2 below shows the primary interface for the Job Sheet used by the JobA, with some mock data for some sample job with the invoice number 0012345-123. On the left are all the primary data fields for the combined Job Sheet and Job Scope Report, and from here the JobA may perform data entry or modification as required.
It is assumed that the Open Job Sheet button and the Save Job Sheet buttons would invoke standardized and familiar GUI combination widgets in the manner of a File Open and File SaveAs, and as such are not illustrated here. The combination file open widget however should incorporate the ability to sort through the list of jobs by: invoice number, client, start date, location, end date and especially by status (being one of Pending, Current or Ended).

The Print Job Sheet button would invoke a standard File Print dialogue and produce a hardcopy or PDF for emailing. The Job Map button invokes a pop-up window illustrated in Figure 5.3.

The Client Contact Details button would invoke a pop-up window displaying the contact information for the client associated with the job (with some possible future functionality to directly place a phone call via some integrated data-voice business telecommunications network solution) illustrated in Figure 5.4. The Job Scope Questions button would invoke yet another pop-up question with a list of prompts to assist the JobA in eliciting required job details (Figure 5.5).

The screen has a configurable element whereby a number of employee placeholders is dynamically created according to the total number of needed employees selected in the widget in the upper right corner. The JobA is able to specify the required Qualification for each (according to information elicited from the client and/or directive given by the MD-Ops/CSO) and also nominate (using a radio button) which employee would be the supervisor for that job.

Each employee placeholder has a pull-down control (shown activated for the second employee) where any special comments or conditions pertinent to that particular placement may be recorded.

Some elements of this form are completed by other roles. The CFO, for example, can complete the invoice number and select a charge rate, or simply instruct the JobA by offline means. The JobA can initiate an invoice number request with the call button beside the invoice number field. The client name, charge rate and job descriptions are all selectable (via a spin button) from pre-made lists created and maintained in other activities. The job start and end dates can be selected using a date-selection widget invoked with the small call-button beside those fields. The job status of PENDING, CURRENT or ENDED is automatically assigned by the system by comparing the start and end dates with the current system date ‘TODAY’.

The JobA does not finalise this form completely, but prepares it (or rather the values of the various instruments it points to) for the employee selection process which is later conducted by the EmpA.

The interfaces given here are by no means claimed to be optimal, let alone elegant, but they are informed entirely by consideration of the activities analysed the instruments declared in the re-design phase.

The claim is that ATSA assembles sufficient data to permit a neophyte to assemble an adequate interface. The very act of analysing the client group’s agenda and doings, then designing a set of data-like instruments, informs a selection of interface
elements, though an experienced UI designer might be expected to have a well developed stylistic palette to draw from.

![Job Sheet for sample job 012345-123](image)

Figure 5.2: Job Sheet for sample job 012345-123

As given below in Table 5.14, the EmpA serves as the subject for pre-re-design activities 17, 18, 19, 20, 21, 22, 25, 26, 27, 32, 33 and 35. Of these, activities 17 to 27 are all associated with the selection of one or more suitable employees for a specific job. The redesign process (as detailed in the attached CD-ROM spread sheet) found that some elements of these could be automated to a degree, but the manner of the business and its agenda require that this crucial set of doings be conducted under experienced human judgement, and open to revision and judgement calls. Accordingly, the UI for the EmpA could allow controls for the pre-redesign activities 17 to 27 to be clustered, and might assume a degree of automation is possible, but must allow user intervention to over-ride automated many decisions and declare the final decision.
As for the JobA’s UI the EmpA would require access to the Job Sheet for a specific job, as illustrated in Figure 5.2. Similarly, the EmpA will need to be able to search for a given job through a combination File Open style widget, described above.

![Image](image_url)

**Figure 5.3: Job Map for sample job 0012345-123**

![Image](image_url)

**Figure 5.4: Client Contact Details for sample client**
As in the case of the JobA, the activity descriptions for the EmpA (given in Table 5.14: Activity Shells for EmpA) predate redesign, and so the list of real instruments declared for that role must also be considered. These are given in Table 5.15 below.

<table>
<thead>
<tr>
<th>ACTY - 17</th>
<th>Selection by Employee default location</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBJECT</td>
<td>EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td>OBJECT</td>
<td>Identify employee(s) closest to the Job location. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td></td>
<td>Travel Distance from (CI 43) GoogleMaps and (CI 4) Employee File</td>
</tr>
<tr>
<td></td>
<td>Select employee(s) from Employee Folder</td>
</tr>
<tr>
<td>RULES</td>
<td>Quick Reference Board accurately records employee’s default location.</td>
</tr>
<tr>
<td></td>
<td>Must identify closest Employee(s) by default location. Closest is defined as smallest travel time. Consult online mapping tools as necessary.</td>
</tr>
<tr>
<td></td>
<td>Must give preference to employee with Job site experience where possible</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[18]</td>
</tr>
<tr>
<td></td>
<td>[NOTE: cache employee ID in an undescribed temporary instrument]</td>
</tr>
<tr>
<td></td>
<td>Record Travel (Commute OR Travel OR LAHA) onto Selection Checklist</td>
</tr>
<tr>
<td></td>
<td>Distance recorded onto (CI 18) Job Booking Sheet for FtgA [24]</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist</td>
</tr>
<tr>
<td></td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td></td>
<td>(CI 6) Quick Reference Board</td>
</tr>
<tr>
<td></td>
<td>(CI 4) Employee File</td>
</tr>
<tr>
<td></td>
<td>(CI 43) GoogleMaps®</td>
</tr>
<tr>
<td></td>
<td>Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>
### ACTY - 18: Selection by Employee Qualification

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator): [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify employee(s) with adequate level of current qualifications. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator): [Current Position: Office Manager]</td>
</tr>
<tr>
<td>RULES</td>
<td>Exclude any candidate without adequate current qualifications. If necessary, repeat earlier checks with relaxed criteria.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[19]</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 18) Job Booking Sheet (CI 6) Quick Reference Board</td>
</tr>
</tbody>
</table>

### ACTY - 19: Selection by Employee availability wrt other Jobs

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator): [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify available employee(s) that have no clashing prior engagements. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator): [Current Position: Office Manager]</td>
</tr>
<tr>
<td>RULES</td>
<td>JobA maintains Current and Pending Job Boards Exclude any candidate with an inconsistent prior engagement.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[20]</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 24) Current Job Board (CI 23) Pending Job Board</td>
</tr>
</tbody>
</table>
### ACTY - 20 Selection by Employee Experience

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Give preference to available employee(s) that have prior experience of the Job site. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>RULES</td>
<td>(CI 33) Employee Experience Register is current and accurate. Give preference to employees with site experience. Where possible, in multi-employee sites, strive to allow less experienced employees to gain site experience in a junior role.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[21] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR returned to earlier check OR Job lost</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 33) Employee Experience Register (CI 4) Employee File Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>

### ACTY - 21 Selection by Employee Medical Status

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify available employee(s) that have current Medical clearance. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>HumA (Human Resources Administrator) : [Current Position: AA]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Read medical status from (CI 4) Employee File</td>
</tr>
<tr>
<td>RULES</td>
<td>Exclude any candidate without current medical clearance. If necessary, repeat earlier checks with relaxed criteria. [NOTE: legal requirements such as Quals and Medical cannot be relaxed] Fail results in immediate change to RED status</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[22] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR returned to earlier check OR Job lost If fail: candidate status changed to RED on (CI 6) Quick Reference Board</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 4) Employee File (CI 6) Quick Reference Board Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>
### ACTY - 22 Selection by Employee Induction Status

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify available employee(s) that have been inducted. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>HumA (Human Resources Administrator) : [Current Position: AA]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Read induction status from (CI 4) Employee File</td>
</tr>
<tr>
<td>RULES</td>
<td>Exclude any candidate that has not undergone induction. Advise (CI 14) FtgA to proceed [24] and CSA [23]</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[23] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR returned to earlier check OR Job lost OR (at CSA’s discretion) candidate may be called in for rapid induction (time permitting) Advise FtgA to conduct FAID check (send (CI 14) Suitability checklist) [24]</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 4) Employee File Unspecified candidate Instrument, listing of candidate employee(s) Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>

### ACTY - 25 Selection by Employee Personal Availability

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Confirm the employee is available according to their personal schedule. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Read (CI 18) Job Booking Sheet Read (CI 4) Employee File contact details</td>
</tr>
<tr>
<td>RULES</td>
<td>Employee must be available and willing to undertake the work.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[26] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR some extra rest time solution determined OR returned to earlier check OR Job lost.</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 18) Job Booking Sheet (CI 4) Employee File Unspecified candidate Instrument, listing of candidate employee(s) Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>
### ACTY - 26  Selection by Employee Agreement to Conditions and Pay

**SUBJECT**  
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**  
Confirm the employee agrees to the payments and bonuses entailed in the Job. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

**COMMUNITY**  
Candidate Employee.

**DivLAB**  
- Read (CI 18) Job Booking Sheet
- Read (CI 4) Employee File contact details

**RULES**  
- Pay rates conform to CFO rates decisions for Client + standard bonuses.
- Employee must be agreeable to the payments and bonuses.

**OUTCOME**  
Candidate Employee(s) identified. [may be more than one] (CI 14)-[27]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra rest time solution determined OR returned to earlier check OR Job lost.

**TOOL**  
- (CI 18) Job Booking Sheet
- (CI 4) Employee File
- Letter, Fax, Telephone, eMail, Personal Conversation etc.
- Unspecified candidate Instrument, listing of candidate employee(s)

### ACTY - 27  Selection by Employee Compliance with Policies

**SUBJECT**  
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**  
Confirm the employee is familiar with, and agrees to comply with, Business Policies on Drugs 7 Alcohol, fatigue and PPE. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

**COMMUNITY**  
- Candidate Employee.
- CSA (Chief Safety Auditor) : [Current Position: MD-Opns]
- JobA (Job Administrator) : [Current Position: Office Manager] [29]

**DivLAB**  
- Read (CI 4) Employee File contact details

**RULES**  
- Employee must conform to policies.
- Employee must have opportunity to declare concerns, request clarifications or declare non-conformance.

**OUTCOME**  
Candidate Employee(s) identified. [may be more than one] (CI 14)-[28,29]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra time for solutions (if available) OR returned to earlier check OR Job lost.  
Record employee compliance.  
Advise CSA and JobA of employee selection.

**TOOL**  
- (CI 14) Employee File
- (CI 14) Suitability checklist
- Letter, Fax, Telephone, eMail, Personal Conversation etc.
- Unspecified candidate Instrument, listing of candidate employee(s)
## ACTY - 32 Update Employee Experience Register: Location and Work Nature

**SUBJECT**
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**
Record details of Employees duties to assist in monitoring and enhancing their experience base. This is necessary so that future employee selections based on experience are accurate. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.

**COMMUNITY**

<table>
<thead>
<tr>
<th>DivLAB</th>
<th>(CI 22) Job Sheet notification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(CI 18) Job Booking Sheet (inc. (CI 20) Job Scope Report)</td>
</tr>
</tbody>
</table>

**RULES**
Job is complete.
The Experience register should be expressed in terms compatible with the Job Scope Request Form (to facilitate matching employees to Jobs)

**OUTCOME**
(133) Employee Experience Register records details of completed Job’s location and nature of work (following consistent taxonomy as Job Scope request form terminology)

**TOOL**
(138) Job Booking Sheet (inc. (20) Job Scope Report)
(133) Employee Experience Register (part of (4) Employee File)

(*)
Note: At the time of the analysis, the Experience Registration Process was a new and still evolving idea. Strongest indication at the time was that the register would be treated as a specific sub-section of the Employee File.

## ACTY - 33 Elicit Supervisor Feedback

**SUBJECT**
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**
Elicit senior (supervising) Employee satisfaction with Employee performance to assist in monitoring and enhancing their abilities. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.

**COMMUNITY**
Senior (supervising) Employee.
CSA (Chief Safety Auditor) : [Current Position: MD-Opns]

**DivLAB**
Employee was places under supervision of a more senior Employee.

**RULES**
Supervisors feedback assists CSA in reviewing Employee status.

**OUTCOME**
CSA advised of Supervisors’s feedback (133). [34]

**TOOL**
(22) Job Sheet
(4) Employee File inc. (33) Employee Experience Register
Letter, Fax, Telephone, eMail, Personal Conversation etc.
Chapter 5 – Method Evaluation

<table>
<thead>
<tr>
<th>ACTY - 35</th>
<th>Update Employee colour Status as required.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBJECT</td>
<td>EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td>OBJECT</td>
<td>Record revised employee colour status in Quick Reference Board. This ensures future selections are accurately decided. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Receive advise (CI 4) from CSA [34]</td>
</tr>
<tr>
<td>RULES</td>
<td>Colour status is a vital indicator for employee selection and the Quick reference board must be kept accurate and current.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Quick reference board (CI 6) updated with revised employee colour status.</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 4) Employee File, inc. (CI 33) Employee Experience Register (CI 6) Quick Reference Board Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>

Table 5.14: Activity Shells for EmpA

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Description</th>
<th>WHERE [Activity] {Role}</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Declared</td>
</tr>
<tr>
<td>addr</td>
<td>employee ADDRESS</td>
<td>[7] {HumA}</td>
</tr>
<tr>
<td>colour</td>
<td>Colour STATUS</td>
<td>[AUTO]</td>
</tr>
<tr>
<td>working</td>
<td>is employee currently working</td>
<td>[AUTO]</td>
</tr>
<tr>
<td>exp-index</td>
<td>(exp-types(exp-index)) = the relevant information for this experience type</td>
<td>[AUTO]</td>
</tr>
<tr>
<td>exp-hours</td>
<td>hours accrued</td>
<td>[..] {HumA}</td>
</tr>
<tr>
<td>qual-alert</td>
<td>Alert message when a Qual is Expiring</td>
<td>[AUTO]</td>
</tr>
<tr>
<td>location</td>
<td>job Location</td>
<td>[14] {JobA}</td>
</tr>
<tr>
<td>qual</td>
<td>Qual per person needed</td>
<td>[12] {JobA}</td>
</tr>
<tr>
<td>distance</td>
<td>distance from job</td>
<td>[AUTO]</td>
</tr>
</tbody>
</table>

Table 5.15: EmpA Instruments

EmpA’s activities 17 to 27 constitute, in effect, a subtractive sorting from a list of all employees, prior to one or more being assigned to a specific job.

Upon opening a specific job, the EmpA will be presented with a listing of all employees in a sortable list. The system will have pre-sorted them by their status colour (green, amber or red) and will have, by default, unselected those with a red status. To reduce screen clutter, the UI as imagined by this author, red-status employees will simply not be displayed, though a button will allow them to be displayed if required (remembering that the CSO has the authority to reclassify employees at any time, and/or to deploy a red-status employee under the supervision of another more trusted employee). Should otherwise unusable employees be displayed, small warning icons will be displayed beside their names.
Figure 5.6: Confirmation Dialogues to prevent illegal assignments

Similar mechanisms allow for the system to automatically hide listings of employees who lack current medical clearance or induction status, though they may be displayed on request as in exceptional circumstances, the CSO may decide to allow them time to become compliant with these pre-conditions if time permits before the job start date. Each of these categories of, otherwise, unusable employees must be flagged for attention to avoid accidental assignment of legally unusable employees.

Assignment is visualised as a click-and-drag from the list in the Employee Selection Tool to the appropriate slot in the Job Sheet window. Any assignment will trigger a confirmation dialogue but the system will automatically display an appropriate warning whenever any unsuitable employee is assigned to a job, highlighting critical issues such as red status, expired medical clearance or a lack of induction. These are illustrated below in Figure 5.6. The dialogue allows the employee record to be amended (under the authority of the CSO) which would then permit assignment. The master Employee List will record the date and time of all amendments to track such alterations.

The Employee Selection Tool (given in Figure 5.7) will present action buttons which open pop-up windows providing further information. By simply highlighting an employee and clicking, the EmpA may view the employee file (with their contact details), the employee’s job history (listing the jobs they’ve been assigned to previously) and their entry in the job experience register.
The tool will sort first by colour status, putting all green level employees at the top, then amber (with red hidden by default). Within that, they will be sorted alphabetically by last name, and then by the distance between their default location (home address) and the job location. This distance will be calculated automatically by a series of automated GoogleMap enquiries conducted by the system using some form of web-service architecture. Should it prove too expensive or too burdensome on network or processor resources, the system may be modified to only calculate these distances upon request. This is a technical issue of implementation, beyond the direct scope of this study. The EmpA may also simply highlight the header of any field and use the Sort button to re-order the listing, as required.

![Employee Selection Tool](image)

**Figure 5.7: Employee Selection Tool**

### 5.6 Conclusion

The novel ATSA method as described in chapter 4, as been applied to a real world small business case. Under to the normative research methodology adopted in chapter 3, and its ideational evaluation, it was necessary to demonstrate that this method could *do* what a systems analysis and design method *can do*, and further, inform a feasible (if not elegant or optimal) attempt at a UI design. Certainly it establishes a possible linkage between the system and UI builders (should they remain in communication via the ATSA specification) such that any variations by one will prompt appropriate changes from the other *within* a common ATSA conception of the design.

The results in this chapter indicate that the ATSA method had indeed elicited, analysed and presented a workable business process description of the subject business. By applying the principles of AT, as expressed in the heuristics and
techniques of the ATSA method, a series of processes were described and then redesigned via the agency of a facilitating computerised system. The ATSA method provided the framework and techniques by which that system was described in such a way as to readily support its implementation.

In final summary, it is the contention of this evaluation that the ATSA method was able to do “what an SAD can do” and thus, under the ideational evaluation approach validated the normative hypothesis that a novel AT method could indeed perform as required and may address some of the challenges identified in chapter 2.
Chapter 6 Discussion of Results

6.1 Introduction

Conducting an evaluation of the Activity Theoretic Software Architecture (ATSA) method against a realistic case study yielded some interesting results. It was necessary to evaluate the method as described at the end of chapter 4, without any reconstruction or re-invention, however through the evaluation process a number of reflections suggested possible refinements and additions.

As discussed in chapter 3, March and Smith (1995) had offered metrics for consideration against each of four classes of research output (Table 6.1, being a reproduction of Table 3.2, presented here again for convenience) lists each of these metrics and criteria against the appropriate research output.

<table>
<thead>
<tr>
<th>ARTEFACT</th>
<th>METRICS/CRITERIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constructs</td>
<td>Completeness</td>
</tr>
<tr>
<td></td>
<td>Simplicity</td>
</tr>
<tr>
<td></td>
<td>Elegance</td>
</tr>
<tr>
<td></td>
<td>Understandability</td>
</tr>
<tr>
<td></td>
<td>Ease of use</td>
</tr>
<tr>
<td>Models</td>
<td>Fidelity with real world phenomena</td>
</tr>
<tr>
<td></td>
<td>Completeness</td>
</tr>
<tr>
<td></td>
<td>Level of detail</td>
</tr>
<tr>
<td></td>
<td>Robustness</td>
</tr>
<tr>
<td></td>
<td>Internal consistency</td>
</tr>
<tr>
<td>Methods</td>
<td><strong>Operationability</strong> (the ability to perform the intended task or the ability of humans to effectively use the method)</td>
</tr>
<tr>
<td></td>
<td>Efficiency</td>
</tr>
<tr>
<td></td>
<td>Generality</td>
</tr>
<tr>
<td></td>
<td>Ease of use</td>
</tr>
<tr>
<td>Instantiations</td>
<td>Efficiency</td>
</tr>
<tr>
<td></td>
<td>Effectiveness</td>
</tr>
<tr>
<td></td>
<td>Impact on environment</td>
</tr>
<tr>
<td></td>
<td>Impact on users</td>
</tr>
</tbody>
</table>

*Table 6.1: Criteria to evaluate artifacts (from March and Smith (1995))*

Having constructed a Method, the metrics and criteria suggested by March and Smith (1995) were those of; Operationability, Efficiency, Generality and Ease of Use. The stated research goals indicated that operationability would be the main criteria. It was necessary for the method to achieve operational functionality, which could be established by a demonstration that it worked and could do what a Systems Analysis and Design (SAD) method can do.

The research framework also dictated that, hopefully, if the method was operationable, and had been built under the terms of the hypothesis (that an Activity Theory (AT) based method could address some of the challenges identified of extant SAD methods) then it could offer some hope for improvement. At the very least, it would demonstrate that an activity theoretic SAD was feasible (a new and potentially valuable contribution in, and of, itself).

First, it was necessary to examine the ATSA method’s claims.
6.2 Claims for the ATSA Method

To examine the claims for the ATSA method, it was convenient to first recap the four steps of the research methodology used in this study per Routio (2004).

1. To identify of challenges.

The challenges identified for extant SAD methods, from Chapter 2, were identified as:

- SAD and Human Computer Interaction (HCI) methods alike tend to lack single, coherent informing philosophical or theoretical bases.
- SAD methods lack clear lines of communication between their phases (however they are arranged, or how tightly or loosely they are defined), to its side-bands such as HCI, and between analyst-designers and stakeholder-users.
- Requirement capture is poorly understood and poorly conducted. There are grounds to consider selecting an informing theory which readily grasps the complexities and scaling issues of user organisations.
- SAD methods tend to lose original focus or coherence and be adapted into expanded and/or ad hoc collections of practice because of high bureaucratic overheads, learnability and/or ease of use issues often resulting from conceptual complexity.
- Current and ad hoc methods in practice rely too heavily on accumulated tacit knowledge and experience, making them difficult to transfer to new entrants and thus discouraging neophyte analyst-designers.

2. To propose a normative hypothesis for improvement against these challenges.

The hypothesis was formed, that: SAD may be improved if conducted through a prescriptive but agnostic method, constructed according to AT principles.

3. To construct a novel activity theoretic SAD method

As exhaustively detailed in chapter 4, such a method was constructed.

4. To evaluate the novel method under a normative case study approach.

Under March and Smith’s (2004) criteria for method-like artefacts produced through research, it was necessary to establish that the novel method could do what and SAD can do.
These four steps, and the results from this research study’s response to them, yields two crucial claims;

1. the ATSA method is novel
2. the ATSA method can do what a SAD method does.

### Claim 1: method is novel

It was claimed that the ATSA method was novel. This was a requirement for the research methodology as discussed in chapter 3.

The notion of addressing some method for collecting requirements and expressing specifications for a system is not novel. The notion of doing so by first examining the business environment is not novel. The idea of concentrating more upon the working roles played by individuals (positions) than by the people themselves is not novel either, having arisen in the Rational Unified Process (RUP)’s Object Oriented (O-O) approach (Krutchten 2004), and being supportable under the OpenPortal presentation framework, which allows for the construction of interface clusters according to the roles played by individuals, a concept known as Identity Based Content Delivery (IBCD) (Dighe 2007, OpenPortal 2009).

The notion of applying AT to the problems of systems analysis and design are also not novel, though the literature shows such efforts have been limited to niche issues such as group interaction analysis, interface evaluation, and specifications for highly interactive web applications.

The novelty of the ATSA method is that it directly addresses the analysis of business and the design of facilitating system specifications at once, entirely under notions of the AT. There has been no need to append elements of Articulation, Distributed Usability or any other principle informing theory. It is not limited to interface design, nor to highly interactive systems (such as web applications), although that was an initial vector of approach taken when building the method.

The use of a single informing theory has meant that each phase in the method is entirely consistent with the others, under the same notions and terminology. Even should ATSA method users specialise and only conduct elicitation, or analysis or design, they should be capable of communicating effectively with those persons conducting the other phases, and also with the stakeholder/users themselves. The group agenda, activity motivations and action goals remain visible and available throughout the method, informing each phase; all design decisions, and indeed providing a solid basis for post-implementation testing and validation.

Further novelty is offered in that the ATSA method offers significant development of notions of interface assembly (the appropriate ‘clustering’ of screen widgets according to use and user) in the same process as it offers detailed specification of the system functionality. This is no side-effect, the two are inextricably linked. If the arrangement of users and their activities were different, a different design would result, and thenceforth to alter one entails corresponding adjustments in the other.
One more point of novelty is the deliberate methodological *agnosticism* of the ATSA method; such that it should remain *not-incompatible with* a broad range of other extant approaches, methods and techniques. This effect keep was intended to keep the method flexible, lightweight and, hopefully, friendly to neophyte users. The same principles, it is hoped, should assist the ATSA method to not self-destruct in the crucible of industrial practice (should it ever be so applied); rather, it should allow a neophyte to produce feasible (if not necessarily elegant) product without years of tacit knowledge and experience, and further allow for modules or components of other methods and techniques to be deployed under the *informing framework* offered by ATSA’s consistent theoretical underpinnings.

6.2.2 **Claim 2: method can do what a SAD method does**

When applied against a realistic case study, it was amply demonstrated that a real world small business could be adequately described and captured in ATSA terms.

Pressman (2001) states that to be able to build a computer system, it is necessary to have sufficient information regarding the inputs, outputs and transformations of data within a system to allow for that system to be built. Whilst this definition is quite generic, and somewhat coloured by the traditional structured approach, it holds.

The realistic case study use of ATSA conducted through chapter 5 resulted in a data-dictionary-like listing of data-like entities which had been described in computer-data terms. The rules and divisions-of-labour had been re-expressed into a pseudo-code-like structured English syntax, and the goals and constraints were retained from the earliest elicitation. Though the realistic case study was conducted in a manner somewhat nearer to a generic image of the traditional structured SAD approach, it is claimed that the resulting specification was sufficiently generic (without sacrificing specificity) that any computer programmer (even an O-O one) *could* comprehend and build from it.

It remains a topic for future work for coders to instantiate designs resulting from the ATSA method and to compare them with the results of other SAD methods. Questions remain as to the relative ease of use for the coder, of a design specified under ATSA; as to the overall speed or efficacy of a software project driven by the ATSA method; and as to any relative merit (size, efficiency, ease of use etc) of the finished system-product *itself* compared with that built for the same client under another approaches. Other future projects were anticipated in which the final phase of the ATSA method could yield specifications more suited to coding paradigms such as O-O.

It must be acknowledged that several components of the ATSA method resembled elements found in other approaches. The RUP in particular, is a well described and documented O-O methodology (though Jacobson might disapprove of that label (Avison and Fitzgerald 2006)). As described by Krutchten (2004), RUP had a number of techniques and notions that sound Much like ATSA: There was a form of scenario-styled walkthroughs of business processes which could identify ‘atomic’ business activities; there was a notion of ‘work-roles’ independent of the individuals who conducted them; there was an emphasis on business process modelling in the early stages; and, the notion of use cases, linking work-roles to system ‘doings’.
It should be noted that the similarities to elements of the ATSA method, found under RUP, whilst conceived under the O-O approach (and thus, arguably, methodological), were not informed by a single coherent theory as to how to understand those interrelated ‘doings’ of the stakeholders organisation. RUP clearly described (through quite a number of interrelated diagramming and modelling techniques) what happens in a business, and even had a notion of capturing actor intention, but had no theoretical framework driving interpretation of the details captured and described. For RUP, as for most SAD methods and approaches, it is sufficient to know what is required.

The argument could be made that if the ATSA method were to be taught in some rapid, lightweight and rather prescriptive way (for neophyte analyst-designers especially) no deep understanding of AT would occur, and any benefit of the underlying theoretical base could be hard to perceive. The use of the ATSA method might then, as with so many other SAD methods, come to rely unacceptably heavily upon the prior experience and tacit skills of the analyst-designer.

As the ATSA method has been constructed from end-to-end entirely upon precepts of AT, all of its tools, techniques, models and terms are consistent and compatible. Even without a formal grounding in AT, a motivating intention was that, at no stage, should an analyst-designer need to translate or transition through or between different modes of thought whilst using ATSA. The chief benefit of using AT was to conduct SAD under a consistent basis throughout. The reader may recall that the initial hypothesis proposed benefits to the use of some theory, and then AT was deemed a candidate worthy of investigation. The hope was that, even were ATSA users to specialise (phase-wise) into ‘elicitors’, ‘analysts’ and ‘re-designers’ working in isolation from each other, each part of their work should be readily comprehensible to all of them. A consistent communicability of terms, notions and of technique should be possible across the method, and, it was hoped, between ATSA users and their clients.

It must be noted that the purpose of ATSA was never intended to teach AT theory to neophyte analyst/designers. That ATSA appears to be operable as a prescriptive sequence of procedures suggests that perhaps the user need not be quite as fully conversant with AT theory as Vrazalic (2004) feared her users would need to be.

By way of contrast, the literature and anecdotal reports suggested that under many extant approaches and methods, as results were produced in one phase or another, they were sometimes effectively translated or transformed into some different conception. Under RUP for example, it might be argued that as the entire vision of the system project becomes re-expressed in terms of its Objects, Classes and Methods, it may then become difficult to unroll (or even communicate) back to an earlier stage of the method/process (say from the engineering back to the requirements elicitation). It remains the subject of future research to quantitatively establish the veracity of such effects, and to test if the ATSA method offers any protection from them.
Chapter 6 – Discussion of Results

6.3 Method or Methodology

Though never a goal of this research study, claimed during the design of the ATSA method nor indicated by the results of its evaluation, the question could have been posed, was there any basis to consider classifying ATSA as a Methodology rather than a Method. In a field where the terms have been used almost interchangeably and the distinction had at times been honoured more in the breach than the observance, it was necessary to consider some definitions.

Whilst dictionary definitions emphasised an original meaning, of or regarding methods (including the study of), they also acknowledged a less distinct newer definition of “a method or body of methods used in a particular field” (OED 2010). For the field of information systems and technology, Maddison (1983) defined an information systems methodology as “a recommended set of philosophies, phases, procedures, rules, techniques, tools, documentation, management and training for developers of information systems.” The holistic nature of such a suite, inclusive of ‘philosophy’ seems to be crucial. Crinnion (1991) added two components: a set of tools, techniques and models together with an overall framework to indicate which tools are to be used at which stages of development.

Avison and Fitzgerald (2006) define an information systems development methodology as a collection of procedures, techniques, tools and documentation tools; deployed in phases; under some philosophical view (the lack of which demotes the collected practices and tools to a ‘method’). So, whilst for Maddison (1983), a set of philosophies may suffice; Avison and Fitzgerald require that the methodology has been developed under theories and assumptions consistent with a ‘philosophy’. They require that a methodology, to deserve the title as such, includes specific reference to its philosophy. As noted earlier, Avison and Fitzgerald would deem RUP a methodology, however much Jacobson might object to the term’s application.

For Checkland (1981) a methodology was “… a set of principles of method, which in any particular situation has to be reduced to a method uniquely suited to that particular situation.” Later Checkland (1985) required that the development of information systems occur under an intellectual framework (philosophy), a methodology and be pertinent for some identifiable and particular application. Checkland’s intellectual framework (philosophy) both guided and constrained analysis and inquiry, providing structure to make sense of the world. A Checklandian methodology was the operationalisation of the intellectual framework through recognition of, or prescriptive to, particular techniques and guidelines.

Certainly ATSA has been constructed under a specific theoretical base, and has maintained a challenge-addressing philosophy from its earliest inception. In its extremely brief life to date, however, it can hardly lay claim to presenting a wide suite of tools and techniques under mature and well articulated prescriptive methods, nor has its best suited particular field of application been identified. In short, ATSA (in the opinion of this author) cannot, as at the time of this writing, lay any defensible claim to the title of a methodology. Hoped remained that, in the face of future development and an adequate base of real-world application, some ATSA-like methodology (with roots in the current work) might identifiably emerge.
6.4 Reuse, Re-Factoring and Re-Engineering

Since the consistent use of an informing theory underlies the structure, terminology and techniques of the ATSA method, it was anticipated that communication between the phases and models should be relatively straightforward. This should entail the ability to move backward through the workflow as well as forward.

The final specification of the user interface widgets, as they appear on each position’s station is determined according to how roles coincide in positions within the client organisation. It is expected that, as the specification of the system occurs largely across activities as conducted by roles, it would be relatively straightforward to unroll the specification of station interfaces, and re-cluster them, according to some new organisational configuration of persons. It was anticipated that, if the group process had been correctly described, in the medium term it should remain largely invariant, as the same doings would need to occur to achieve the group agenda. There would be no need for a complete re-examination. Though not directly tested in the chapter 5 evaluation, this effect was foreshadowed in earlier publications (Brown et al. 2005).

It was not unreasonable to speculate whether deeper reuse or re-factoring (in support of re-engineering) might be feasible under the ATSA method. If the nature of a doing should vary to some noticeable extent, the analysis could unroll back even to the elicitation phase. It should be relatively straightforward, in a conceptual sense at least, to reverse engineer any decisions made in the design phase, re-factor the underlying doings, and re-engineer a new specification.

The re-engineering of the models and of the specifications they generates, however straightforward under ATSA, may not prove simple in the final code of the instantiated system. Two vectors to address this are anticipated. Firstly that, in some future work, the ATSA method is adapted so that its design phase identifies object-like and class-like entities (with instrument transactions suggesting methods), such that O-O engineering approaches may be adopted for instantiation. Secondly, interface assemblages might be semi-automated through the use of frameworks similar to that of the OpenPortal standard (Dighe 2007, OpenPortal 2009).

6.5 Unexpected Results

As discussed in chapter 4, during the struggle to extract a prescriptive method from the famously powerful but slippery notions of AT, it was constantly tempting to stray towards deploying other concepts and/or mechanisms. It came as a pleasant surprise that whenever the construction of the method fell into a dead-end, the answers arose from an examination of the precepts of AT itself.

It was also unexpected that the ATSA method should produce such promise of broad applicability. Initially motivated by the Standish Group and OASIG reports of unacceptably poor success rates, and a personal leaning towards interface studies, early research assumed that studies of usability may provide a key to increasing system acceptance rates (and thus, hopefully, overall success rates). As described elsewhere, early work concentrated on repatriating Vrazalic’s 2004 usability evaluation method.
Late in chapter 4, during construction of the method, the “penny dropped” that ATSA had the potential to operate as a genuine SAD method. Briefly the focus shifted away from the interface, but it became apparent that (equally unexpectedly) a strong shaping specification for the interface clustering could be produced in the same process as the specification of system processes, functions and data entities. Indeed, the two seemed inextricably linked.

The possibility should be acknowledged that a sense of potentially ‘broad applicability’ for the ATSA method may be the result of two factors: that the method is immature and barely tested in any quantifiable manner (particularly in practice), so its limitations are unclear, and; its very agnostic and deliberately flexible and lightweight descriptions might simply mean that it appears initially to be not incompatible with most other approaches. Further detailed research into linking ATSA to other approaches and techniques may reveal that it had serious limitations unseen at the time of this writing.

6.6 Reflections of Practice

It was important that reflective comments upon the experience of having applied the ATSA method, as described in chapter 4, against a realistic case study be set out and considered.

6.6.1 Improved Template Tools

During the earlier chapter 4 feasibility test, the ATSA method seems to have provided a clear concept for the analyst-designer to follow however the details of the real world test case in chapter 5 were as inconsistent and incomplete as one might expect. The genuine, organically evolved real world small business was chaotic in its structures and processes to the point of near dysfunction. This provided considerable challenges for the analyst-designer. The open and non-specific ATSA tools proved flexible enough, but challenging; given that they (naturally) lacked any mature directives and examples to guide deployment and application.

There emerged considerable scope for the development of template style elicitation and analysis tools to assist an in rephrasing real world doings and utterances into ATSA’s terms. Future work might be indicated, towards the development of supporting tool sets, perhaps including semi-automated Computer Assisted Systems Engineering (CASE) tools.

6.6.2 Hierarchic Roles, the assignment of authority

As described in section 5.3.2, the business examined in the evaluative case study had chosen to assign a three tier hierarchy of authority to its roles. Purely functional roles were designated ‘Administrative’. Decision making roles were assigned the status of ‘Officer’. Final authority and corporate responsibility were vested solely in the owner operators, and their roles were designated as ‘Directorial’.

The ATSA method, as described in chapter 4, did not specifically anticipate a differentiation of roles by authority (a notion of organisational ranking). It was hoped that the ATSA method’s requirement to record deontic relationships however would entail as much detail as required. There is scope for future work to more carefully examine the issue of accurately capturing hierarchies among and between roles.
6.6.3 Enhanced Activity Shells

A further enhancement to ATSA tools was suggested from the experience of conducting the realistic case study evaluation. It was considered possibly useful to have some manner of quasi-automated analysis CASE-style tools. Once initial elicitation has identified roles and candidate Instruments, it could be helpful to be able to simultaneously view lists of these whilst assembling graphs or tables of transactional linkages.

Some form of a natural, gestural interface could allow for the simple drawing of transactional links between nodes, with automated inheritance of the new relationships recorded against the lists of instruments and roles. Indeed, the Engström-styled activity shells could be at least partially auto-populated by way of these simple gestural point-and-draw interactions.

A computer enhanced tool of this form, probably best instantiated via some technology similar to those seen on recently popular multi-touch personal electronic devices, could be complimented by an extension to the information contained in the Engström shell, so that it could record a notion of ‘sequencing’. As seen in chapter 5, the form of the Engström activity shell recorded Instrument transactions, but sequencing information had to be derived from temporal constraints recorded during analysis.

<table>
<thead>
<tr>
<th>ACTY - 26</th>
<th>Selection by Employee Agreement to Conditions and Pay</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBJECT</td>
<td>EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td>OBJECT</td>
<td>Confirm the employee agrees to the payments and bonuses entailed in the Job. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>Candidate Employee.</td>
</tr>
</tbody>
</table>
| DivLAB    | Read (CI 18) Job Booking Sheet  
Read (CI 4) Employee File contact details |
| RULES     | Pay rates conform to CFO rates decisions for Client + standard bonuses. Employee must be agreeable to the payments and bonuses. |
| OUTCOME   | Candidate Employee(s) identified. [may be more than one] (CI 14) [27]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra rest time solution determined OR returned to earlier check OR Job lost. |
| TOOL      | (CI 18) Job Booking Sheet  
(CI 4) Employee File  
Letter, Fax, Telephone, eMail, Personal Conversation etc.  
Unspecified candidate Instrument, listing of candidate employee(s) |

Table 6.2: Activity Shell 26, from Appendix G

A simple format enhancement to the activity shell could allow for a series of horizontal cells, read from top to bottom, horizontally divided between Division of Labour (DivLAB) (inward) and outcome (outward) directions. Separation into the horizontal lines would simply represent sequence, and simultaneity could be captured by the sharing of a cell or line.
Chapter 6 – Discussion of Results

For example, in case study activity number 26, (as seen in Appendix H, reproduced below for convenience as Table 6.2) the activity shell was given as:

In this instance, the sequencing of the transactions was not immediately obvious. A simple tabular addition could represent sequence in the manner of Table 6.3 (below)

<table>
<thead>
<tr>
<th>IN (DivLAB)</th>
<th>OUT (Outcome)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instrument 4 (from node 25)</td>
<td>ENQUIRY (to employee)</td>
</tr>
<tr>
<td>AGREEMENT (from employee)</td>
<td>Instrument 4 (to node 27)</td>
</tr>
</tbody>
</table>

Table 6.3: Some Sequenced Transactions from Activity 26

6.6.4 An Expanded Hyper-CAT

As previously observed in Chapter 5, the Combined Activity Table (CAT) represents an adjacency matrix of the activity network. The CAT and the network are equivalent and each may be produced readily from the other. Whilst a Network diagram proved impractical for the realistic case study examined in chapter 5, Appendix I (below) presents the CAT for the test case.

Whilst the CAT listed the thirty five activities examined during the evaluation, a number of transactions (such as the storage or retrieval of employee files in the employee folder) were most easily represented by presenting a vague entity ‘Folders’ on the CAT. Likewise ‘Boards’, ‘Cache’, the external groups ‘Employees’ and ‘Clients’ and the external 3rd party softwares ‘MYOB®’, ‘FAID® and ‘Online Banking’ were shown as entities capable of receiving or sending candidate instruments.

Consideration of such peripheral non-activity entities could have indicated the potential to bend of transactions through the yet-to-be-described system. Some utility in slightly expanding upon the ATSA method might be indicated.

Capturing, with enhanced detail, peripheral entities of this type and their transactions may serve to allow ATSA designed systems to be more readily “woven in” module-wise with other systems (especially if they are also described in some ATSA-friendly way). In this manner, an hierarchically modular deployment of ATSA can be envisaged, where larger systems might be designed as linked sub-systems, each designed according to the agendas of organisational units. Mosaics of multiple ATSA-styled systems might best be represented in some abstracted hyper-CAT which represents whole activity networks (systems) as nodes that exchange Instruments.
6.6.5 Node Reduction Heuristics (NRHs)

As set out in chapter 4, the analyst-designer is encouraged to apply some node reduction heuristics towards the redesign of the business process and the specification of the system. Specifically, the designer is to consider identifying ‘pipes’ and ‘joints’.

Pipes are where an instrument passes to and from the same actor, whilst joints are where an instrument passes through an activity unchanged. According to the heuristics, activities which had the characteristics of a joint, and were one end of a pipe transaction were strongly indicated candidates for collapse, re-organisation or re-configuration.

However, in practice, it proved not to be quite so simple. While the heuristic holds in principle, in application some complexities arose. What was inferred but had not been stated categorically was that while the Node Reduction Heuristics (NRHs) effect Activities, they are conducted on a ‘per-candidate-instrument’ basis (looking at the transactions of pre-instrument documents, files etc), and they examined the candidate Instruments’ transactions and transforms. Pipe and joint characteristics might have proven resolvable by converting the candidate instrument to a ‘real’ data-entity ‘system’ instrument, or simply by more carefully examining what was really happening during the transformations and transactions.

As stated in chapter 4, the NRHs work best when the instrument is simple, and when the activity has small numbers of transaction pathways running through it. In the real case, some candidate instruments were non-simple, potentially comprised many ‘real’ instruments. Some of the ‘real’ component instruments may well have proven ‘collapsible’ (which is to say, would not actually have survived to become ‘real’ Instruments) whilst others may not collapse, but become instantiated as data-like-entities in the ultimate system specification.

A number of observations were made by the analyst-designer during the conduct of the real world case study, and they were recorded here:

- Whilst the CAT and other models were recorded in a tabular form (after the manner of adjacency diagrams, as allowed under the method as described in chapter 4) the analyst-designer expressed a desire for a graphical representation of the activity network. Specifically, there was a desire for some form of dynamically re-arrangable network diagram with automated inheritance capabilities.

- An Activity with many instrument transactions may have been a Joint, or one end of a pipe, for one or more of its instruments, but might not have been be for some others. This suggested that some of the activity could be split away and collapsed, but not the whole of it.

- The analyst-designer envisaged the ability to move an activity node and then see its linkages move with it. More importantly,
there was a desire to have the ability to split or pare-away some subset(s) of a node creating new nodes, but with an automated inheritance of linkages and constraints. Such a tool, ideally, should also have the ability to check for ‘unseen strands’ linking the ‘pared-away’ instrument to others from the original ‘parent’ activity’s transactions.

- If some component parts of an activity were to be collapsed then it would be necessary to check if the remaining transactions may have required access or ‘exposure’ to that collapsed Instrument. It should be remembered that any instrument may be a component for some larger multi-Instrument transformation. This type of inherited obligation was not improbable in the case of a read-type action where a compiled report (virtual or physical) was required. When candidate instruments (essentially just business documents) became real Instruments (that is data objects transacted between stations and the system across the interface) then doings such as a read-type action would ultimately give rise to another real Instrument (a report or dialogue display perhaps) which would present just the data values from the ‘document’, perhaps required for some larger multi-Instrument transform.

Ultimately, it seemed that the NRHs were not invalidated, but their scope and power will require some refinement and re-examination. They may well be less likely to locate and remove a redundant activity (though that is still possible) but seemed to be a better indicator for the need to examine a candidate Instrument’s transactions and transforms to assist in determining if it should be redefined, resolved (and/or split) into ‘real’ instruments (leaving, perhaps, some component available for collapse). It should be offered, for clarification, that the identification of ‘real’ instrument is one of the crucial goals of the design phase of ATSA, as they become the data-like-entities of which the system is comprised. It the NRHs assist in that process, they remained a valuable part of the method.

Not unlike some surgical procedure, the paring away of fascia and other extraneous ‘non-functional but connective’ material must be done with great care lest one accidentally sever an important connection. The NRHs served an important role in identifying zones that would require more careful work during redesign

6.7 Conclusion

Having discussed the results of the realistic case study evaluation, and some reflections and unexpected results, it now remained to offer a final conclusion to the thesis, together with consideration of the limitations of the research study and an indication of potential future work based on this thesis.
Chapter 7  Conclusion and Future Work

7.1 Conclusion Statement
This research study aimed to produce a usable Activity Theoretic systems analysis and design method. The goal was to explore if Activity Theory (AT) could serve as the underlying principle for an entire method. An analysis of the literature suggested that the lack of an informing theory underlying many extant methods and techniques may be a contributing factor to issues of poor communication, poor usability and poor understanding of user requirements which had been cited as elements of the widely accepted low success rates for computer systems.

The challenges identified in chapter 2 were that:

- Systems Analysis and Design (SAD) and Human Computer Interaction (HCI) methods alike tend to lack single, coherent informing philosophical or theoretical bases.

- SAD methods lack clear lines of communication between their phases (however they are arranged, or how tightly or loosely they are defined), to its side-bands such as HCI, and between analyst-designers and stakeholder-users.

- Requirement capture is poorly understood and poorly conducted. There are grounds to consider selecting an informing theory which readily grasps the complexities and scaling issues of user organisations.

- SAD methods tend to lose original focus or coherence and be adapted into expanded and/or ad hoc collections of practice because of high bureaucratic overheads, learnability and/or ease of use issues often resulting from conceptual complexity.

- Current and ad hoc methods in practice rely too heavily on accumulated tacit knowledge and experience, making them difficult to transfer to new entrants and thus discouraging neophyte analyst-designers.

The seeming lack of consensus regarding approaches to, and applications of, software development methods suggested motivation towards assembling the framework for a methodological approach informed by a consistent and coherent theory, which could prove useful, at least for some classes of software system.

A suitable theory was required to underpin this new method, and the seeming taxonomic trend towards activity orientation suggested an approach. The literature suggested AT could have value in User Interface (UI) design. Application of AT had also been attempted in a systems design context, predominantly for highly interactive web-applications. AT might therefore, have been the best fit as the informing theory.

To have identified the challenges, a possible vector of approach to them, created a novel method along that vector, and to have demonstrated that such a product could feasibly operate as systems analysis and design methods do, is herewith offered as a
sufficient, significant and new contribution to the domain. It remains in the realm of future work to quantitatively evaluate a mature version of this method against other extant approaches.

Having surveyed the non-trivial array of tools, techniques, methods, methodologies and approaches to be found in the fields of software and systems analysis and design (including that of their interfaces), this investigation found some grounds to question if the absence of a single coherent informing theoretical framework might contribute positively to the field. Such a principle, which by some definitions is a prerequisite for use of the title ‘methodology’, might enhance communication between various lifecycle phases and by thus ameliorating miscommunications increase the reportedly poor success rate of systems.

Further hypothesising that a close attention to user needs could enhance acceptance rates for systems, a user-centrism was adopted. Any informing coherent theoretical base should therefore be one that prominently features target users.

It was further observed that analysis and design methods often appeal (if only by omission) to the tacit knowledge of skilled and experienced analysts and designers. It was hypothesised that a single coherent theoretical base might produce a prescriptive method suitable for neophyte analysts and designers. Such a method might not approach the elegance of expert solutions, but might at least enable to neophyte to produce feasible solutions with confidence. Such a method could also, perhaps, remain sufficiently agnostic of extant approaches, that it could serve as a useful pedagogical tool, and evolve in the hands of its user to accommodate other methods, techniques and tools; ultimately serving as a something of a methodological framework.

As the rather heterogeneous field of HCI design methods was of considerable prior research interest to this author, and itself a variant (or subset by some accounts) of systems analysis and design famous for its user centrism, initial investigation centred on the notion that perhaps a single theory could be applied to HCI. A review of relevant literature uncovered a history of attempts to apply AT to HCI where interest in its potential was first observed (Martins 2007). The literature recognised that the precepts of AT certainly ought to be deployable and at the least, it was hailed as a promising a useful framework for establishing contexts in end-user work environments (Quek and Shah 2004).

The nearest and most recent methodological HCI work featuring AT at the outset of this research was that of Vrazalic (2004) which described an interface usability evaluation method. Deploying both AT and the principles of Distributed Usability (DU), Vrazalic’s method was to establish what a given system would be used for, and to then evaluate its usability towards that end. Initial investigation questioned if the AT driven identification of a system’s end use might have greater utility earlier in a system’s analysis and design lifecycle (or at least, that of its interface).

The driving motivation was to explore if AT could serve as the single coherent theoretical base on which an analysis and design method could operate. It was decided to construct a method closely following the principles and components of AT as given in its third generation, that of Engström (1999). A normative evaluation
research methodology (Routio 2005) was adopted to determine if an innovative new method, built specifically to address specific issues, could feasibly work.

Perhaps not unexpectedly, the construction of an innovative AT based method during this research proved difficult in the extreme as the precepts of AT are famously difficult to tie down and deploy prescriptively. After numerous false starts, documented throughout chapter 4, a set of techniques were arrived at which unexpectedly showed promise of applicability beyond HCI. Indications were that this innovative Activity Theoretic Systems Architecture (ATSA) method (as it was named) might serve for the analysis and design of information systems in general, inclusive of basic requirements for their user interfaces. Thus, having come full circle, AT showed indeed promise as an informing coherent basis across which systems analysis and design might be conducted.

The ATSA method was designed in strict accordance with the hypothesis by adhering to the precepts of AT and with a view to addressing issues of coherence, communication and ease of use by the inexperienced.

The ATSA method, as described in chapter 4, was applied by this author against an anonymised real world small business. Throughout chapter 5 and further detailed in the appendices, the ATSA method indeed seemed to allow this author to analyse the users’ requirements and specify in some detail a system, in terms which a coder might readily instantiate. In this way, functionality was indicated; that is to say: The novel ATSA method was able to produce essentially the same result as a current SAD method, being in this case according to Pressman (2001), sufficient information regarding the inputs, outputs and transformations of data within a system to allow for that system to be built. This satisfies the notion of ideational evaluation as described by Iivari (2002).

Under the terms of the normative research methodology, since the testing of the ATSA method against a real world case indicated functionality, the hypothesis is supported. The claim is thus defended; that AT can serve as a coherent theoretical basis for systems analysis and design, inclusive of its interface requirements.

7.2 Limitations

It must be acknowledged that some limitations applied to the ideational evaluation in chapter 5. Firstly, the test case was a small business, with a small set of individuals. It should be noted that the processes of this business were non-trivial and included quite complex interactions. It can be asserted that if ATSA could cope with these, then it could cope with the processes of many larger business units. It remains unexplored if ATSA would cope with a large number of such units in combination. Further work may be indicated in exploring the efficacy of an ATSA approach to large organisations on a unit-by-unit basis, perhaps linking them by their system-to-system Instrument transactions. The flexible scalability of the AT precepts (best seen in Leon’tev’s work (1978, 1981)) may lend itself quite well to such an hierarchic up-scaling.

Secondly, the business processes under consideration were highly interactive, with the stakeholders operating in close personal contact. A larger organisation might
spatially distribute its members and operators in such a manner that many interactions are already mediated by some informational or computational technology. Though such a case was not tested here, it is the contention of this author that the precepts of AT anticipate the use of ‘mediating tools’ of any sort (electronic, manual or psychological) and can describe, analyse and allow for re-design irrespective of extant ‘systems’. Some future research could …

Thirdly, the redesign of business process was deliberately ‘gentle’ striving quite deliberately to preserve the culture and flavour of the existing business. The ATSA method did not, of itself, provide any direct guidance towards any radical business process re-engineering, though this author would content, it provides the familiarity and descriptive tools to facilitate discussion with stakeholders towards that end. Future work might investigate the applicability of AT precepts, as expressed in the ATSA method, for supporting significant re-design of business processes.

The ATSA method, as presented and evaluated in this thesis, whilst striving to remain somewhat agnostic of extant approaches, methods and techniques, tended more strongly towards a traditional structured approach. Without directly referring to any of the classic techniques of the structured approach, ATSA produced something sufficiently like a process-oriented data-dictionary as to be comfortably deployed under traditional implementation by coders trained in that paradigm.

Some readers might deem it a limitation that this study offered no quantifiable evaluation of the efficacy of ATSA compared to extant methods and approaches.

A significant set of limitations arise from ATSA’s somewhat restricted adoption of AT’s component concepts. The design of ATSA chose quite deliberately to set aside some of the more complex elements of fully developed AT (see section 2.6.5). This means that ATSA is limited in the extent to which the full richness of AT may inform design decisions. Notably, ATSA does not involve itself in the subtleties between objet and predmet (Kaptelinin 2005) or the object of desire (Natdi 2005) and the socio-political tensions involved within groups as activity driving objects are formed.

As set out in section 2.6.5, ATSA chose to freeze a snapshot of the doings of a group, and then strive to facilitate these doings. This means that the rich dimensions of cultural-historical changes are not fully utilised. As discussed in chapter 2, Uden et al. (2008), in embracing the cultural-historical development of activities, hoped to design navigational interfaces to embrace and facilitate the temporal flexibility of human activity. ATSA is limited in its ability to embrace this notion, as it (significantly) exemplifies an alternate use of AT; to deliberately reduce the burden of complex group doings (effectively reducing them to actions and operations), thus liberating the humans to explore more rewarding notions.

As described in section 2.8.8, Mwanza-Simwami (2001, 2002, 2009) deliberately embraced the notion of AT contradictions and used them to assemble a clever set of design-informing questions. In not embracing these contradictions (indeed, in striving to remove them) ATSA may be limited in how much it benefits from this rich set of data.
Though this author has a trained academic background in the socio-cultural embeddedness science and technology (BA Hons in the History and Philosophy of Science, UOW), and although major figures in the AT literature such as Bannon and Bodker (1991) do not appear to fully grasp the notions either (see section 2.6.7), it must be conceded that the construction of ATSA followed an rather deterministic approach and thus, in its current form, is limited by that decision.

A final limitation is thus suggested, that the ATSA method has not been demonstrated as doing “what an Object Oriented (O-O) or Agile SAD might do”. To have addressed these (and other) significant and competing paradigms would have been well beyond the scope of this research. It remains for future work to explore how the ATSA method might interface with other approaches such as O-O or agile. It is the contention of this author that by staying deliberately non-aligned when building this method, it should not have become unfriendly towards any of them. Some avenues of approach to these other approaches might already be anticipated, perhaps by replacing Universal Modelling Language (UML) activity and usecase diagrams with AT representations, by searching for potential identifications of O-O ‘objects’ and ‘methods’ when examining instruments and their transactions, and even by developing rapid and lightweight AT-flavoured models deployable under the Agile Modelling (AM) approach.

7.3 Future Works

As with any research output which has made a contribution, there should have been some immediately obvious vectors for possible future work. Indeed, the presence of such possibilities might be an indicator of contribution, or at the least, least of novelty. Regardless, no research project is ever truly completed, and there should always be some scope for ongoing tests or investigations.

In the case of the ATSA method, being a novel method among a great many methods (and indeed debates as to the need for any methods at all), numerous vectors for possible future work are immediately apparent, a number of which suggest bodies of work extensive and innovative enough to readily occupy some future doctoral thesis projects (which this author would greatly enjoy supervising in some capacity).

A convenient initial framework for presenting these vectors, as foreshadowed in chapter 6, is to address the remaining three of March and Smith’s (1995) suggested evaluative metrics for method-like research outputs.

7.3.1 March and Smith’s Other Method Evaluation Criteria

As described previously in chapters 3 and 6, March and Smith (1995) offered four evaluation metrics to be considered in testing the claims of “method-like” research outputs. These were: operationability, efficiency, generality and ease of use. Chapter 6 discusses how the ATSA method has satisfied the criteria of operationability, in that it could demonstrably do what an SAD can do through the realistic case study conducted in chapter 5.

Though not strictly required under the research framework selected for this study, some future work is justified to establish even stronger evaluation of the ATSA
method by addressing the other three of March and Smith’s (1995) suggested evaluation metrics. These were: efficiency, generality & the ease of use.

These serve as a convenient framework for presenting some of the vectors for possible future research work arising from this current study.

7.3.2 Efficiency

As discussed briefly in chapter 2, some commentators (notably Tom Gilb (2006)) are very earnest in their demands to see quantifiable evidence comparing any method’s claim. To provide an adequate test of the ATSA method against other extant approaches however would require both that ATSA had matured to the point that there was a body of practice and of practitioners, and an amount of analytical work worthy of a significant research project in its own right. Neither of these would have been feasibly within the scope of the current research project.

A future project would be, once some level of use has occurred (‘maturing’ ATSA to a stage where quantifiable measurement and comparison would be meaningful) to conduct a series of rigorous tests. There is no small issue to be addressed as to how one evaluates the merits of a method. Gilb’s own commentaries (2005) recognise the fundamental difficulties in conducting any such test. To date, many such claims are largely experiential or arise from surveys of analyst-designers and/or stakeholder-users projects in the real world (each of whom has experienced a project through a single approach or method).

As well as identifying appropriate aspects of SAD methods to test, it would be necessary to assemble a research methodology and experimental procedure to conduct meaningful tests. Perhaps some double-blind testing of the end results of systems built under different approaches to meet the same scenario. Issues might include: the degree to which stakeholders can meaningfully participate, the ease with which coders can work to the specifications supplied, how readily the end product can be validated or verified against those specifications, how quickly the project ran, how many man hours were required, how usable the end product was for the user and whether the final product would be deemed a success or not. It might also be meaningful to test the learnability of the method for neophytes (so that the degree of prior experience is not a meaningful skewing factor in the tests).

It would also be no small issue to quantifiably evaluating the relative benefits of ATSA over other SAD methods, and simply establishing how any one method’s benefits can be identified and measured against another’s would be a non-trivial exercise and itself make a contribution to the field.

7.3.2.1 Real World Validation

Quek and Shah (2004), in reviewing five pre-2005 activity based information systems development methods, found that only Mwanza’s (2001) Activity Oriented Design Method (AODM) eight-step approach showed real world validation, and three others (Korpela et al.’s ActAD (1997), Kaptellinnin et al.’s checklists (1999), and the Martins and Daltrini framework (1999)) showed some small scale experimental testing.
Although the chapter 5 evaluation of the ATSA method already approaches the degree of real world validation found in 2004 for Mwanza’s AODM. Further validation against real-world systems is indicated.

7.3.3 Generality
To explore March and Smith’s (1995) third suggested metric for the evaluation of method-like research output; that of generality, future work could explore establishing linkages between ATSA and other systems analysis and design approaches, methods and tools.

Several possible lines of future investigation are briefly described below.

7.3.3.1 BPM
During the construction of the ATSA method in chapter 4, it was observed that, should the AT-based method under construction yield a feasible AT flavoured Business Process Modelling (BPM) method, it would be a worthwhile outcome.

Numerous extant approaches contain, entail and/or make use of the modelling (mapping) of business processes. BPM is a worthy commercial and academic pursuit in its own right. The International Organisation for Standards’ ISO 9000 (2008) suite of Quality Management standards, and most especially their ISO 9001 standard for a Quality Management System (QMS), have become de rigueur necessities for businesses seeking higher market standing and access to some classes of government funded work. The ISO 9001 standard requires that business process be fully documented and described, including descriptions of what work is performed, in what way, by which persons.

There is some scope for an ATSA styled analysis to yield adequate information to assist greatly in meeting some of the requirements of the ISO 9001 standard. Future work is indicated to explore this possibility, and there is considerable scope for commercialisation.

7.3.3.2 Object Orientation
As presented in chapters 4 and 5, the ATSA method leans somewhat towards more traditional structured (indeed procedural) design paradigms. This was a deliberate choice to reflect a wide audience of practitioners, a broad range of skills and the authors own shortcomings as a coder. It must be acknowledged that a large, industrially active and healthy community of Object Oriented (O-O) practitioners and their tools and methods exist.

Only the final stages of the redesign phase of the ATSA method, as shown in chapter 5 however, turned towards structural design notions. Once activities, Roles and Transactions have been identified and their pathways re-designed, there remains considerable scope for the identification of instruments and their transactions as O-O styled classes, objects and methods.

The ATSA method was constructed to remain deliberately agnostic of extant approaches, and it is claimed that ATSA should remain non-unwelcoming of other approaches and conceptions. Indeed, it has been observed repeatedly through this
thesis that several notions contained under ATSA’s AT-informed phases bare a close resemblance to notions found in some forms of O-O design.

A significant and entirely necessary filed of future projects emerging from this thesis encompass the best ways to mate an ATSA method to O-O notions of design, coding and engineering. There is no fundamental reason why they should be antagonistic or incompatible. Indeed, ATSA may yet yield a pathway towards some notion of reconciliation between those methods deemed ‘procedural’ and those deemed ‘data-centric’.

7.3.3.3 UML
It would be remiss of this thesis to ignore the massive and highly significant contribution to practice of the UML suite of standards, models and tools, most especially activity diagrams and use cases. As noted above, the ATSA method (despite its entirely different point of origin) unintentionally echoed numerous ideas found under UML. This should be seen as evidence that these ideas themselves have some inherent value, if multiple vectors of approach reach non-incompatible conclusions.

Supported by the Object Management Group (OMG), a consortium of many large and influential entities in industrial practice, UML will not be ignored. Some, such as Bell (2004) may query its sometimes over-enthusiastic but uninformed adoption, but when used under a methodological approach its ubiquity, efficacy and communicability has rarely been questioned.

There is considerable scope for linking ATSA to UML. At times, in some applications, UML might be seen to lack an informing principle (contributing to misapplication) and it may be that ATSA can point the way towards resolution. At the very least, should the ATSA method show some promise for industrial application, then it had best find a way to dress itself in UML clothes, or face considerable barriers to adoption.

7.3.3.4 Scenario Based Design
There have been a number of design approaches which attempt to examine and capture the richness of dynamic business processes over time by conducting some form of ‘walk through’. Often these postulate ‘what if’ questions (scenarios) and explore a proposed system’s response to the sequence of requests and interactions generated under that scenario.

The Rational Unified Process (RUP) is compatible with a product called the Unified Scenario Based Design (IBM 2005). The description of that product includes the interesting observation that, “modelling a system with only use-case driven UML specifications does not allow a good level of business reactivity” (IBM 2005).

Whilst ATSA in the form described at the end of chapter 4 in this research project is in no way a functional scenario-based modeller, the creative narrative-based notion of Story Fragment Analysis (SFA) seen in chapter 4 (one of several false starts towards the building of ATSA), could be seen as something of a scenario-based concept.
There is scope for future work, in which the concept of an AT informed SFA is revived, linked with the ATSA method, and used to generate a coherent scenario modeller. It is anticipated that an activity based scenario notion could not only offer some insights towards system design, but generate viable and realistic test-cases against which to validate built systems against their AT informed specifications.

### 7.3.3.5 Formal Methods

As mentioned in chapter 4, there is considerable scope for future work in formalising the handling of the temporal and deontic constraints which appear inside goal and motive statements whilst analysing business processes under the ATSA method.

It is anticipated that if such work could be done, considerable progress would result towards relating the ATSA method to Formal Methods. If this could be achieved, it might serve to assist Formal Methods in handling the largely social and human ‘non-functional requirements’ that currently provide difficulty.

It may prove useful to develop formal protocols to handle: the inheritance of constraints by Single Action Nodes (SINs), actions and activities under abstraction and refinement; the representation and logical calculus of temporal and deontic constraints under some multi-modal logic; and, the satisfaction of constraints under redesign (including consistency checking).

### 7.3.3.6 Service Oriented Architecture (SOA) and Web Services

Under new notion of Service Oriented Architecture (SOA), small bespoke systems and applications could be seen as outdated under a cloud-computing paradigm. The principle of web services offers lightweight modules of functionality, available on-the-fly through the internet, to be assembled as required for larger purposes.

Though much debated and still somewhat controversial (for reasons of deliverability and security) the future for an ATSA flavoured SAD Method might be questionable in the face of this paradigm.

Though this research did not address itself to SOA or web services, it is conceivable that the ATSA method (which elicits, analyses and described units of doing), could readily aid in identify web-service functionalities to which an ATSA-built system might subscribe; and/or could just as readily yield identify functionalities which the system could make available for sale as a web service to other systems in the cloud. A considerable amount of future work to explore SOA, web services and the possible applicability of the ATSA approach is indicated.

### 7.3.3.7 User Interface Design (HCI)

The field of HCI and the design of usable interfaces, whilst characterised in this research study as a ‘side band’ of a larger domain encompassing the building of computer systems, must be recognised as a mature, and thoroughly academic discipline in its own right. Issues of interaction studies, usability engineering and assessment and of ergonomics are just a few of the many identifiable sub domains within this important field.
Chapter 7 – Conclusion and Future Work

Despite this author’s prior vocational experience, interest and academic studies within the fields of HCI and the UI, this research project produced outcome which; though remaining cogniscent of UI matters, focussed more on the more general analysis and design of whole systems.

The ATSA method, however, in the form presented in chapter 4, indicates what sorts of interaction are required during which work modes, and can further specify how these functions should best be clustered into work stations; but remains silent as to the actual appearance and operation of individual interface widgets and layouts. At the time of this writing, the ATSA method hopes to provide a somewhat architectural, or structural, specification within which the classic skills of UI designers might best select modes of implementation. In this, the ATSA method’s degree of specificity to interfaces is not unlike that it offers for the system’s coders; whose skill in implementation and instantiation is still required.

There are entire branches of possible future work in exploring how the ATSA method might build closer ties to the HCI/UI domain, perhaps enhancing or complimenting UI design to a finer degree.

7.3.4 Ease of Use

The fourth of March and Smith’s (1995) evaluative criteria addresses ease of use for method-like research outcomes. The reader may recall that many extant SAD approaches were challenged by issues of learnability. Some were seemingly hampered by poor transmission of principle and technical execution. One hope for the ATSA method was that, by being based on a single consistent informing theory throughout, it should have a consistency of concept facilitating learnability (even if the somewhat complex underlying theory itself were only presented as a gloss).

At the time of this writing, the ATSA method is undergoing a significant qualitative test. A sizable cohort of over 150 second year undergraduates in the School of Information Systems and Technology (SISAT) at the University of Wollongong (Australia) are being instructed in the use of the ATSA method, alongside the theories and techniques of extant SAD and HCI methods. The students are receiving approximately 10 hours of direct instruction and approximately 20 more hours of laboratory practice. Their primary assessment task, in small groups of three or four, is to present design specifications for small business systems (scenarios for a number of which have been carefully prepared with degrees of complexity only slightly less than that presented in chapter 5 of this thesis).

Under the authorised approval of the University of Wollongong’s human research ethics review process (application reference HE10/045, approved March 4th 2010), this author and others anticipate publications to present anonymised excerpts of student work and reflective commentaries, in the evaluation of ATSA’s learnability and applicability to neophyte analyst-designers.

It was hoped that, if this experiment proved successful, a number of these students might adopt ATSA (or some elements of it) in their final year annual project studies, where they’re required to design and specify a system in detail, usually for some real world client organisation. Already at the time of this writing, one student project group (one of whom was learning the ATSA method with the 2nd year cohort
mentioned above), had formally requested to apply the ATSA method for their real-world/real-client final-year annual project. It was hoped they would do so, would achieve some usable result, would inspire others to do so; and would make their experiences and reflective feedback available.

A more ambitious hope for the ATSA method was that, if it should prove readily usable by neophytes, it could facilitate production of at least feasible systems designs by inexperienced analyst-designers. If this could be demonstrated, then ATSA, or some future (possibly more prescriptive) methods based upon its concepts and example, could allow early career industrial entrants to produce acceptable and workable product prior to gaining decades of experience. Of course the author expects that an experienced analyst-designer could produce a more elegant solution, and perhaps cut some procedural corners, but if an ATSA-styled approach could allow the neophyte to be more successfully productive sooner, then the industry as a whole could benefit from a greater number of more experienced analyst-designers being made available for future software projects.

Even if the ATSA method were to prove itself a useful purely as a pedagogical tool, a ‘sandbox SAD teaching method’ as it were, that would prove a significant contribution, quite aside from any measurable impact it may have on the success rates of software projects, or the easing in to productivity of neophyte industrial entrants.

7.3.5 Other Potential Future Works

Beyond those future works suggested by consideration of March and Smith’s (1995) evaluation metrics for method-like research products, other vectors for possible future work arose during the conduct of this research project.

7.3.5.1 Validation and Verification

Under system validation, the analysis and development of system specifications leads “to a complete and consistent specification.” (Lakos and Malhotra 2002). Verification ensures that the system has complied with the specifications. These two complimentary checking processes (together commonly referred to as V&V) comprise a non-inconsiderable body of literature and academic research.

Since the ATSA method generates specifications to suit the work needs of a group’s process, under their agenda, the notion of seeking validating for those specifications is entirely justified. Future work is indicated, exploring the best techniques and approaches towards offering verification for ATSA specifications. There may be scope for linking such future work with that of exploring Scenario Based Design, following up from the currently discarded notion of Story Fragment Analysis (SFA), as described above.

Verification of the final systems built from ATSA-derived specifications and checking such systems’ compliance with the ATSA specifications it was built to meet, would be expected in the marketplace for ATSA to be treated as any serious option. As indicated previously, the SFA notion, perhaps coupled with some elements of Vrazalic’s (2004) DUEM principles, may prove a useful starting point for future research towards this end.
Chapter 7 – Conclusion and Future Work

7.3.5.2 CASE Support Tools
Numerous, often commercial, Computer Aided System (or Software) Engineering (CASE) tools have been produced to facilitate software and system creation under various extant approaches and methods. Should the ATSA method, under further development, extension and testing, prove some promise in real world application, then the development of some CASE tools for it would not be contra-indicated.

There are a bewildering variety of CASE tool types available (Avison and Fitzgerald 2006) including drawing tools, modelling tools, project management tools and integration tools offering databases of meta-data.

Should it prove justified by real world interest in the ATSA method (or any extensions or refinements upon it), then future work could be undertaken to construct various CASE tools. Just three particular CASE-like tools were immediately indicated by the authors experience in conducting the evaluative realistic case study, though numerous more might be readily imagined.

7.3.5.2.1 Elicitation Database
To assist in the elicitation of positions, roles and candidate instruments (which can be a non-trivial exercise, requiring some iteration and cross-checking with the stakeholders), it seemed it could be useful to have a lightweight and portable database tool.

![Figure 7.1: An Early Attempt at a POSITION Table](image)

Figure 7.1 to Figure 7.4 show some very primitive first-pass attempts to construct some relational database tables and input forms to assist in conducting the ATSA method. As described in chapter 4, such exercises served more to assist conceptually in the final building of the ATSA method itself than in its early applications. They reflect a period in the building of the method before the workflow had fully formed and this method’s tactics for navigating Leon’tev’s (1978, 1981) hierarchy had been expressed. Ultimately then, these tables and forms were not employed in this research study, though they serve to illustrate the notion of a CASE-like database tool.

260
Chapter 7 – Conclusion and Future Work

Figure 7.2: An Early Attempt at a ROLE Table

Figure 7.3: An Early Attempt at a CANDIDATE INSTRUMENT Table

Figure 7.4: An Early Attempt at an ACTION Table
7.3.5.2.2 Graphical Support for Analysis and Redesign
As was raised in chapter 6 (above), assembling the activity network with constant reference back to lists of roles and of instruments was a non-trivial challenge for the analyst, and so it was conceived that some form of a natural, gestural interface could allow for the simple drawing of transactional links between nodes, with automated inheritance of the new relationships recorded against the lists of instruments and roles. The activity shells themselves (such as those presented in Appendix H) could perhaps be created and even partially populated by way of these simple gestural point-and-draw interactions.

The design and specification of a highly interactive graphical tool of this type would, of itself, prove an interesting project for the application of the ATSA method. To specify the functionality and interface of this sophisticated CASE tool would require the investigation of a single position group, with multiple (if closely related) roles. Such an interactive system would require careful specification for its UI, and that alone would challenge the fine-grained applicability of ATSA principles.

7.3.5.2.3 Semi-Automatable Interface Assembly
In the final stages of the ATSA method, once specifications have been assembled for activities and the roles which conduct them, station interfaces can be described, wherein clusters of screen sets can be made available for positions, according to the roles they embody.

CASE-like tools for the rapid, and perhaps semi-automatable assembly of these Stations could provide considerable aid. The ongoing use of a system designed under ATSA, particularly in the face of some organisation re-arrangements (whereby different clusters of roles may be instantiated in reconfigured positions) could benefit from some CASE-like station management tool, which could permit a system administrator to rapidly and easily re-factor and re-engineer the ATSA-designed system to accommodate. Perhaps products such as the OpenPortal presentation framework (donated to the OpenSource community by Sun Microsystems), which already allows for the construction of interface clusters according to the roles played by individuals, a concept known as Identity Based Content Delivery (IBCD) (Dighe 2007, OpenPortal 2009), could be incorporated into an ATSA CASE tool towards this end.

7.4 Final Conclusion
There is significant scope for future works, which of themselves promises some future and possible longevity for the ATSA method, if only as the first of a wide family of AT approaches to SAD and UI design. It is hoped that, to have finally demonstrated some genuine feasibility for AT as an informing theoretic principle in system design, other activity-centric approaches may emerge or be re-invigorated. Hopefully, some positive steps towards redressing some challenges which continue to face computer systems analysis and design, and that of their user interfaces, will result. The ATSA method is herewith presented for consideration, testing, development, teaching and, hopefully, application.
References


References


264


References


References


References


Henderson-Sellers, B. (2006), Keynote Address and personal conversation Advocatus Diaboli Symposium, ENASE’05 Erfurt, Germany.


References


References


Maciaszek, L. and Liong, B. L. (2005), Practical software engineering : a case study approach, Harlow, England : Pearson/ Addison Wesley


References


Oxford English Dictionary [online] available from


References


Popper, K. (1934), *The Logic of Scientific Discovery*, [translated from the German to English in 1959 by the author], Routledge.


References


Vrazalic, L. and Gould, E. (2001), Towards an activity-based usability evaluation methodology, in Anders, Mørch and Opdahl (eds.), *Electronic proceedings of 24th information systems research seminar in Scandinavia (IRIS24)*


## Appendix A: Glossary of Acronymns

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACD</td>
<td>Activity Centred Design</td>
</tr>
<tr>
<td>ActAD</td>
<td>Activity Analysis and Design</td>
</tr>
<tr>
<td>AM</td>
<td>Agile Modelling</td>
</tr>
<tr>
<td>AODM</td>
<td>Activity Oriented Design Method</td>
</tr>
<tr>
<td>ASD</td>
<td>Amethodical Systems Development</td>
</tr>
<tr>
<td>AT</td>
<td>Activity Theory</td>
</tr>
<tr>
<td>BCS</td>
<td>British Computer Society</td>
</tr>
<tr>
<td>BDUF</td>
<td>Big Design Up Front</td>
</tr>
<tr>
<td>BIS</td>
<td>Business Information System</td>
</tr>
<tr>
<td>BPMN</td>
<td>Business Process Modelling Notation</td>
</tr>
<tr>
<td>BPR</td>
<td>Business Process Re-Engineering</td>
</tr>
<tr>
<td>CASE</td>
<td>Computer Aided System Engineering</td>
</tr>
<tr>
<td>CFO</td>
<td>Chief Finance Officer</td>
</tr>
<tr>
<td>CIO</td>
<td>Chief Information Officer</td>
</tr>
<tr>
<td>CLE</td>
<td>Constructive Learning Environment</td>
</tr>
<tr>
<td>CSCL</td>
<td>Computer Supported Collaborative Learning</td>
</tr>
<tr>
<td>CSO</td>
<td>Chief Safety Officer</td>
</tr>
<tr>
<td>CSCW</td>
<td>Computer-Supported Cooperative Work</td>
</tr>
<tr>
<td>CTP</td>
<td>Cambridge Technology Parters</td>
</tr>
<tr>
<td>DUEM</td>
<td>Distributed Usability Evaluation Method</td>
</tr>
<tr>
<td>EIS</td>
<td>Executive Information System</td>
</tr>
<tr>
<td>ERP</td>
<td>Enterprise Resource Planning</td>
</tr>
<tr>
<td>GAO</td>
<td>United States Government Accounting Office</td>
</tr>
<tr>
<td>GDD</td>
<td>Goal Driven Design</td>
</tr>
<tr>
<td>HCD</td>
<td>Human Centred Design</td>
</tr>
<tr>
<td>HCI</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>High-IQ</td>
<td>High Interaction Quotient</td>
</tr>
<tr>
<td>HTA</td>
<td>Hierarchic Task Analysis</td>
</tr>
<tr>
<td>i*</td>
<td>Distributed Intensionality</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines</td>
</tr>
<tr>
<td>IPA</td>
<td>Internal Plane of Action</td>
</tr>
<tr>
<td>ISO</td>
<td>International Organisation for Standards</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technology</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>NCC</td>
<td>National Computing Centre of Great Britain</td>
</tr>
<tr>
<td>NFR</td>
<td>Non-Functional Requirement(s)</td>
</tr>
<tr>
<td>OASIG</td>
<td>Organisational Aspects (of IT) Special Interest Group (UK)</td>
</tr>
<tr>
<td>OMG</td>
<td>Object Management Group</td>
</tr>
<tr>
<td>O-O</td>
<td>Object Oriented</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RE</td>
<td>Requirements Engineering</td>
</tr>
<tr>
<td>RUP</td>
<td>Rational Unified Process</td>
</tr>
<tr>
<td>SAD</td>
<td>Systems Analysis and Design</td>
</tr>
<tr>
<td>SATBPA</td>
<td>System for Activity Theory Business Process Analysis</td>
</tr>
<tr>
<td>SD</td>
<td>Strategic Dependency</td>
</tr>
<tr>
<td>SE</td>
<td>Software Engineering</td>
</tr>
<tr>
<td>SCOT</td>
<td>Social Construction of Technology</td>
</tr>
<tr>
<td>SDLC</td>
<td>System Design Life Cycle</td>
</tr>
</tbody>
</table>
Appendix B: Listing Business Company Procedures

The specialist labour hire Company used for the evaluating Case Study had, at the time of the evaluation, documented many of its core business activities. For reference, these are presented here, purged of extraneous headers and of any direct identification of the business itself.

As described in chapter 5 a representative subset of these processes was taken to run the ATSA Method against, in order to evaluate its potential and feasibility as a SAD method.

The business procedures documented at the time of the evaluation were as follows:

**Advertising for Employees**

- Check Managing Director determines the need for advertising
- Determine the most appropriate newspaper or magazine on a cost, distribution and location basis
- Determine the number of advertisements and the duration of the advertisement.
- Check with Managing Director if the standard advertisement wording is suitable.
- Proceed to book the advertisement
  ELSE
  - Generate a new wording proposal.
  - Get approval from Managing Director for the new wording.
  - Proceed to book the advertisement.

**Frequency of and appearance of advertising**

- The business may advertise safe-working positions in various newspapers around.
- Advertisements of this kind MUST NEVER specify a pay rate.

**Response to Advertising**

- Phone calls will be received from people in response to the advertisements run in the various newspapers.
- When someone calls, obtain the following information:
  - Name
  - Address
  - Contact Number
  - Qualifications
- Where they obtained our details (i.e. newspaper, internet, etc.)
- To assist the business in determining whether the advertising has been effective, it is also recommended that we obtain which newspaper the caller saw the ad.
- You can also advise the caller that the form can be downloaded from our internet website.
Appendices

- Advise the caller that we will be sending out an information pack which they need to return with colour copies of their tickets, drivers licence and category 1 medical report
- If an applicant contacts the business by some other means, such as email or post, then respond and attempt to establish a telephone conversation. As well as following the process described above, enquire where they heard of the business, and specifically ask if it was:
  - Website
  - Business card
  - Word-of-mouth (from whom)
  - Other …

**Outputs**

- Standard recruiting advertisement
- Purchase Order for ad placement in one or more newspapers &/or trade papers
- Brief applicant details – name, contacts, qualifications

**New Employee Application**

- Upon receiving a phone call/eMail/letter or fax from any applicant, confirm their contact details and send out an employment Information Pack.
- Place a copy of the employment enquiry letter to Employment Enquiries Folder
  - record the persons contact number, qualifications (i.e. tickets) and any other relevant information
  - attach copies of all required documents
  - attach a blank Reference Check Sheet
  - attach a blank Employment Suitability Form
- This pack will comprise of the following documents:
  - Employment Enquiry Letter (Word template)
  - Employment Application Form (PDF file)
  - ASU membership form (option information for employees)
  - Superchoice Letter (Word Template)
  - Superannuation choice form (PDF file)
  - Information Sheet
- Check that the applicant has noted they hold all appropriate PPE
- The employment enquiry letter will advise the person to return the documents to the office address.
- Place a copy of the employment enquiry letter in the employment enquiries folder and write on the copy the persons contact number, qualifications (i.e. tickets) and any other relevant information.

**Employment Application received**

- Upon receiving an application in the mail, check the employment enquiries folder for the copy of the letter & attach to the application. Check that copies
of the required document are attached to the application and attach a reference check sheet for reference checks to be conducted.

- If any documentation is missing, contact the person & request the missing information. If they are waiting on someone else to provide the information, note on the employment enquiry letter & check back with them in no more than a week’s time.
- Check that the person has noted that they hold all the appropriate PPE’s

**Tax Free Threshold Declaration and Superannuation Details**

- As part of the application pack, each applicant is issued with the appropriate forms to specify:
  - If they wish to claim the Tax Free Threshold for their income
    - If an applicant FAILS to return a tax declaration, the EmpA will issue a REMINDER by phone, email or fax (etc).
    - Without this declaration, the business is legally obliged to apply full taxation to their payments, and if they may be able to claim back any excess at the end of the financial year.
    - If an applicant DOES return a tax declaration form, claiming the tax free threshold, then the office of the business will:
      - Advise the Australian Tax Officer (ATO) and RECORD the date on the original (file) copy of the tax declaration form
      - Make the appropriate adjustments to the employee’s details in the MYOB program, and RECORD the date on the original (file) copy of their tax declaration.
  - Their preferred Superannuation Provider
    - If an applicant FAILS to return a superannuation preference, the EmpA will issue a REMINDER by phone, email or fax (etc).
    - Without this, the superannuation contribution from the business will be paid into an account managed by its default preferred provider.
    - If an applicant DOES return a declaration of preferred superannuation, then the office of the business will:
      - Advise the nominated superannuation provider and take all necessary actions, and RECORD the date onto the original (file) copy of the employee’s declaration.

**Outputs**

- Employee File
- Acceptance Letter
- Rejection Letter
- Welcome Pack
- Employment Suitability Form
- Entry in Employee Suitability Quick-ref List
- Entry into MYOB System
- Entry into FAID System

285
Appendices

- Record of ATO date and MYOB date on their tax free threshold declaration.
- Record of Superannuation provider date on their superannuation declaration.

Marketing procedures

- Website
  - The business maintains a corporate website to advertise and explain our service.

- PR Presentations
  - Senior officers of the business (most especially the Managing Director Operations) will make personal presentations of our services to senior officers of external organisations (including rail transport operators) which may work in the rail corridor.

- Business Cards
  - Officers of the business carry a supply of corporate business cards and are encouraged to present these to any and all interested persons
  - Officers of the business are encouraged to collect business cards from all persons associated with companies that have dealings in the rail corridor, to expand our list of contacts
    - Business Cards are to be stored in the Business Card Folder
    - Contact details are to be entered into the contact phone list, sorted by company/organisation.

- Tenders. Federal and State Government works are allocated by tender, and many of these are publicly announced on the internet or by email.
  - The office of the business will monitor public tender announcements of work conducted within the rail corridor.
  - Companies named in such tenders will be contacted if they are considered potential clients – ie: if the business could provide worksite protection services.
    - Schedule a cold-call in the Office Diary.

- Cold Calling
  - The CLO (assisted by the AA) will regularly place courtesy-calls to established and potential client organisations to see if we could supply them with worksite protection.
    - All established clients are to be called on a regular basis (roughly every two weeks)
    - Newly identified potential customers are to be called at the first available opportunity.
  - All outgoing calls are to be logged in the office call register.
  - If the potential client has no work for the business, then request if we may call again in a few weeks – and schedule a call in the OFFICE DIARY.
  - If the cold call produces a direct request for service, then:
    - If the customer is an established client, proceed to the job booking procedure
    - If the customer is NOT YET an established client follow the new client procedure.
If a cold call produces a request for information – then an information package must be sent by whatever method best suits the potential client.

- **Cold Call eMail**
  - The CLO (assisted by the AA) will place unsolicited eMail notices to selected clients, soliciting work.
    - Consider the client’s sensitivity and responsiveness to this mode of communication, AND ALSO
    - Consider the current business head office internet usage.

**Outputs**

- Client list.
- Business card register.
- Client contact list.
- Call Register entry.
- Office Diary entry.

**New Client Establishment Procedure**

- Commence a new client application form – recording all relevant information
- Request credit references and conduct a credit reference check for the prospective client.
- Establish a rates schedule for this client.
- Create a Client File.
- Create a MYOB entry for online transactions and invoicing.

**Outputs**

- Client list.
- Business card register.
- Client contact list.
- Call Register entry.
- Office Diary entry.
- Credit Reference Application (CRA) form.
- Client File.
- MYOB entry.

**General Job Booking Procedure**

- Contact the client
  - The Client Liaison Officer (CLO) will contact the client at the first available opportunity
- Confirm pay rates
  - Only the MD-Admin is authorised to create, discuss or negotiate pay rates with clients.
  - The Client Liaison officer (CLO) may conduct pay-rate related discussions ONLY if authorised, directed and instructed by the MD-Admin
Appendices

• Initiate Job Booking Sheet
  o Invoice Number
    ▪ This is left BLANK – until after the job has been completed. When the MD-Ops enters the job into MYOB, it will generate an Invoice number, which is then recorded onto the Job Booking Sheet
    ▪ For very large scale jobs which may engage numerous employees or run over extended periods, then MORE THAN ONE Job Booking Sheet might be initiated.
    ▪ These will all be collected under ONE SINGLE Invoice Number
    ▪ This situation is to be AVOIDED whenever possible.
  o Employee details
    ▪ Once the selected employee(s) are confirmed then record the details onto the Job Booking Sheet
  o Client details
  o Job Details
  o Employee Checklist
    ▪ During the employee selection process, the employee is asked directly about their fitness to work and familiarity with their obligations. These are recorded in the Checklist on the Job Booking Sheet.

• Elicit Job Scope
• Initiate Employee Selection Procedure
• Preliminary FAID© fatigue check
  o The FAID© checkbox is ticked AFTER the preliminary Fatigue check has been conducted, in an effort to ensure that the employee(s) will not be unduly affected by fatigue during this job placement.
• Post Job Booking Sheet onto the “Pending” Jobs Board
  o When the date of the job occurs, the Job Booking Sheet is moved to the “Current” Job Board

Outputs

• Job Booking Sheet
  o Record the ‘order taken’ details
• Posting on the Pending Job Board
• Preliminary FAID© assessment

Job Booking – Job Scope Procedure

• Upon receipt of a Job request (in any form) the office of the business is to make every reasonable effort to elicit as many details of the job as possible, most especially:
  o Exact Location (consider travel time or LAHA)
  o Duration (consider fatigue effects)
  o Start Time (consider need to attend pre-work briefings)
  o Expected end time
  o Number of persons in work party
  o Physical extent of worksite (consider mode of protection needed)
o Transport/communication requirements
o Nature of the work being conducted

- Whenever possible, send the requesting client a Job Scope Enquiry Form and request they return it as quickly as possible, with as much detail as possible
- Ensure a senior business Protection Officer (and/or the MD-Ops) has the opportunity to review the information.
  o Contact the client to confirm details or seek clarification, as required.
  o Contact the client to advise the necessary level and mode of worksite protection, and the costs rates) involved.

- CLIENT OPERATOR
  o Client Operators generally prefer to liaise directly with the *principle contractor* for each job. They leave it to this contractor to engage *ancillary* services (such as worksite protection). The Client Operator therefore does not issue Job Scope details to the secondary contractors (such as the business) and the Principle Contractor generally seeks to minimise their costs by specifying only the absolute minimum degree of safety protection.
  o When a Client Operator directly engages the services of the business, it is generally easier to elicit Job Scope details.

- CLIENT
  o As described above, the business will make every reasonable effort to elicit Job Scope details from

- PRIOR INSPECTION by a business PO
  o the business will offer the services of a senior PO to visit worksites in advance whenever possible (the client WILL be charged for this service unless an MD decides otherwise)

- OFFICE RESEARCH
  o Google maps
    ▪ The internet service “Google Maps” offers the ability to examine aerial photos of most sites.
    ▪ the business office will ‘call-up’ the location of proposed worksites on Google Maps and attempt to ascertain as much information as possible from this (such as location of nearest roads, shops, medical facilities etc)
    ▪ If the job is confirmed, the worksite protection staff may be forwarded a copy of the Google Map image(s) if deemed useful/necessary.
  o Client technical maps
    ▪ The office of the business has access to published client technical maps. Whenever necessary or deemed useful, the business worksite protection staff will be issued with extracts from these maps to assist them in understanding work sites.

**Outputs**

- Job Scope Reports.
- Decisions regarding the appropriate level of protection for a specific job.
- An information package to be sent to the worksite protection staff assigned to the specific job.
Appendices

- Employee Experience Register
  - Once fully developed, this register will record work types & conditions.

Job Booking - Employee Selection Procedure

Consult Job Booking Sheet & Job Scope

- The client will have provided details of the job including:
  - Location and duration
- Based on the client’s request and/or the assessment of a business safety assessor, the number and rank/grade of required worksite protection personnel will be determined.
- The job scope should indicate any specific job requirements or conditions, to be considered when selecting an employee.

Identify Candidate Employee

- It is necessary to identify a suitable candidate employee to be assigned to the job.
- This person must be evaluated for their suitability before confirming them in the job.
- It is necessary to identify which candidates could possibly fill the client’s job request. There are two main factors to consider:
  - Employee FITNESS for WORK
  - Employee location, relative to the work.
- Fitness for work will always take precedence over location.
- A person who cannot satisfy ALL of the fitness requirements cannot be assigned to rail corridor duties.

Check Location

- Consult the Quick Reference Sheet and give preference to potential employees who live closest to the work area.
- The business recognises that clients will always prefer to engage a local employee because they:
  - (a) wish to minimise their travel/LAHA (Living Away from Home Allowance) costs and
  - (b) prefer to use an employee with local work site familiarity.
- The business is always sensitive to client (customer) requirements, and wherever possible will endeavour to engage a local, or the candidate with greatest proximity to the worksite.
- The business will also endeavour to track the work experience of its employees, and give preference to candidates that have previously worked in the job area.
- When a job requires multiple employees, the business will attempt to ensure that at least one has site familiarity (if possible) and will attempt to give employees lacking familiarity with that site the opportunity to become familiar.
Check Status

- Check Quick Reference Board
  - Is candidate GREEN? IF yes Proceed
  - ELSE Is candidate AMBER? IF yes try to replace with a GREEN status candidate OR use only for low priority jobs where supervision is available wherever possible strive for seek opportunities to IMPROVE candidate’s capabilities with mentoring seek feedback on performance from client and/or senior hand after each use successful deployment may prompt STATUS upgrade review unsuccessful deployment may prompt STATUS downgrade review
  - ELSE Is candidate RED? IF yes try to replace with a GREEN or AMBER status candidate OR use ONLY in most exceptional cases and only under supervision feedback required from client and/or senior hand after each use successful deployment may prompt STATUS upgrade review unsuccessful deployment may prompt STATUS downgrade review ELSE select a new candidate OR Consider requesting a change of scope or of timing by the client. Consider telling the client we CANNOT fill the job.

Check Job Boards

- Check Pending Job Board and Current Job Board
- ensure selected candidate will be available at the time of the job, AND
- ensure selected candidate will not be fatigued because of other jobs and/or travel
  - IF both yes, Proceed
  - ELSE Select a new candidate OR consider changing assignment. Ensure no existing clients job is disrupted or disadvantaged
  - If any employee is re-assigned from another job, then repeat this entire procedure for all jobs that are affected and now require an employee.
  - It is a priority that ALL jobs effected in this way are filled BEFORE any new jobs are accepted or actioned.

Select Candidate

- Having identified the most likely candidate for a job, it is necessary to confirm that the person is suitable and able to conduct the work assignment.
- Confirm FITNESS FOR WORK
  - The business must take every reasonable step to ensure FITNESS for WORK.
  - Persons who are unfit for a unit of work will not be assigned.

Confirm Fitness of Selected Employee

- Having identified a likely candidate for the Job, based on their availability, location and status, it is absolutely necessary to ensure their FITNESS FOR WORK
  - Fitness for work entails several factors
Appendices

- Adequate competency tickets
- Have satisfied medical background checks
- Has been inducted into the business policies & procedures
- Is clear for Fatigue

**Check Tickets**

- From the Employee Folder, check that the employee holds CURRENT tickets of the appropriate level, and for the appropriate rail network.
  - If the Job requires the holding of a Power Out Permit (POP), then ensure the employee is properly qualified to hold a POP.
  - Without this, the person MAY NOT BE PLACED, and a new candidate must be selected.

**Check Experience**

- Once the “Experience Register” process has been finalised and implemented
  - Check that the employee has either:
    - Prior experience of the work conditions indicated by the Job Scope OR
    - Has been judged ready (by a Managing Director) to conduct work of the type described in the Job Scope. This is typically done to:
      - Increase the employee’s experience base
      - Enhance the abilities of our pool of employees.

**Check Cat-1**

- From the Employee Folder, check that the employee holds a CURRENT Category-1 Medical Certificate.
  - Without this, the person MAY NOT BE PLACED, and a new candidate must be selected.

**Check Induction**

- From the Employee Folder, check that the employee has been indicted into the policies and procedures of the business.
  - Without this, the person MAY NOT BE PLACED, and a new candidate must be selected.

**Check FAID©**

- The FAID© program is used to check fatigue levels in advance of Job allocation, and is used to record actual hours worked after each Job.
  - To assist in assessing the selected candidate’s fatigue status, enter the hours of the Job against the candidate employees FAID© entry and check to see if they will not be unfit for work by reason of fatigue, as assessed by the FAID© program.
  - Check FAID© to ensure the candidate should NOT be fatigued if assigned to this job
Appendices

- IF yes, Proceed
- ELSE select a new candidate

Consult Employee

- Records and files can only indicate so much. They may have other potentially fatiguing activities the business is unaware of, other time commitments, they may be sick or they may not be sure that they will be free from the effects of drugs, alcohol or fatigue etc.
- The following points details the MINIMUM level of consultation necessary before assigning an employee to a Job.
  - speak with the candidate to confirm their current circumstances and condition.
  - There may always be unforeseeable changes to the candidate’s circumstances, capabilities and qualifications and this consultation is an opportunity to check for these.
    - Any report of significant variations MUST be recorded and may require an update of the employee file, their entry on the Quick Reference Sheet, and/or fresh copies of CoC’s, the Category-1 Medical declaration or other documents.

Consult Employee re Availability

- The employee must be asked about their availability to work the Job
  - Make allowance for travel time.
  - Ensure the employee has the means and time to travel safely and legally to the Job.
  - Consider the fatigue effects of any other work they may have immediately prior to, or following the Job.
  - Check travel distance and time by consulting http://maps.google.com
    - This free internet search tool generates a standardised distance and travel time estimate and may be independently confirmed by candidates, clients or others.
  - Negotiate travel and/or LAHA allowances (consult with Managing Director (Administration) for ANY variations from the standard initial offering)
    - As necessary, consult the client. If necessary, there may need to be adjustments.

Consult Employee re Pay Rate, Travel & LAHA

- Confirm the pay rate, any travel allowance, LAHA (Living Away from Home Allowance) or other allowances or conditions.
  - If the employee is not prepared to work for the pay and allowances offered, then select another candidate.
- It is necessary to NOTE that the Employee has AGREED to the reimbursement package prior to them commencing the Job.
  - This agreement will be formalised in the Job Letter, which is sent to the employee.
Appendices

Consult Employee re D&A, Fatigue, PPE

- It is necessary to provide the employee with the opportunity to SELF DECLARE any concerns that they may have about their FITNESS TO WORK
  - The business actively encourages employees to monitor and self-declare their fitness for work.
  - No employee is ever to be punished for having Self Declared a concern
  - The Employee MUST BE ASKED, and MUST DECLARE, that they are aware of the requirement to be free from the effects of Drugs or Alcohol.
    - They must declare that they WILL be free from the effects of Drugs or Alcohol.
    - If they choose to Self Declare a concern, they may NOT BE PLACED on the Job, and A B Safety Rail Pty Ltd’s Drug and Alcohol Policy is to be consulted for the appropriate follow-up actions to take.
  - The Employee MUST BE ASKED, and MUST DECLARE, that they are aware of the requirement to be free from the effects of Fatigue.
    - They must declare that they WILL be free from the effects of Fatigue for the Job.
    - If they choose to Self Declare a concern, they may NOT BE PLACED on the Job, and A B Safety Rail Pty Ltd’s Fatigue Policy is to be consulted for the appropriate follow-up actions to take.
  - The Employee MUST BE ASKED, and MUST DECLARE, that they are aware of the requirement to have and use all appropriate PPE (Personal Protective Equipment).
    - They must declare that they have and will use all appropriate PPE for the Job.

- After these details have been confirmed, they must be CHECKED OFF on the Job Booking Sheet

Outputs

- An appropriate employee has been selected and assigned to the job
- The employees name is entered into the Job Booking Sheet, and it is posted to the Pending Job Board
- Job Letter
  - With attached Job Scope details, it is sent to the employee
- ‘preliminary’ FAID© check entry
- Client is consulted regarding any variations to the initially agreed rates and allowances.
- Job Booking Sheet
  - Tick the Job Filled box
  - Tick the Details Advised box
  - Tick the FAID box (for the preliminary FAID check)
Job Booking - Confirm Employee Procedure

- Contact the client and advise them of the name and rank (level of worksite protection qualification) for the employee(s) selected
- Employee Contact details will be provided to the client so that they can liaise with the employee to update details and job requirements.
  - If a client refuses the services of a specific employee:
    - Enquire the reason why
    - Advise the MD-Admin
    - A review of the employees employment suitability classification may be in order
    - The employee may need to be consulted
    - If there has been some underlying (previously unreported) incident or irregular occurrence, then the MD-Admin will undertake the appropriate action – in accordance with the business’ Incident & Accident policy.

Outputs

- “client confirmed employee details” section of the Job Booking Sheet will be completed.
- Any variations in the needs or wants of the client must be actioned by the MD-Admin
  - Reported difficulties with specific employees may require investigation.

Job Booking - Job Confirmation Procedure

- Contact the client
  - Final confirmation of Job and Employee details, especially:
    - Location
    - Dates & times
    - Work type
    - Employee name, contact & rank
    - Any special requirements
    - Charges and fees
- Contact the employee
  - Final confirmation of Job details, especially:
    - Location
    - Dates & times
    - Work type
    - Fitness-for-Work
    - Any special requirements
    - Pay rates, conditions and allowances

Outputs

- Call register or email-sent folder records contact with Client and Employee(s)
- Client confirmed: emp details section of the Job Booking Sheet completed
Appendices

Job Check Form Procedure

- The CLO will issue clients with a blank copy of the Customer Satisfaction form and forward (or fax) it to selected clients.
  - If necessary, a follow up phone call may be required.
- Any issues, problems or complaints are to be actioned by the MD-Admin, who may instigate further procedures to address the matters.

Outputs

- Job Completion check recorded on the Job Booking Form
- Completed survey form
  - To be shown to the MD-Admin
  - To be filed in the client’s folder

Customer Satisfaction Procedure

- A customer satisfaction survey form will be printed off from the corporate network template folder and sent by post or fax to a selected client.
- A phone call may be placed to the client to advise them, and/or to elicit feedback.
- Any problems identified in the client feedback may generate remedial action at the direction of the MD-Admin. This may require the instigation of further procedures.

Outputs

- A customer satisfaction form created and dispatched
- Job Completion check recorded on the Job Booking Form
- The complete customer satisfaction form will be stored with the client’s file.

Job Progress Check Procedure

- The CLO will contact the client (usually a supervisor or other person with knowledge of the worksite conditions) to check:
  - That the work commenced as planned
  - That the worksite protection employee(s) attended on time and in proper order
  - That work is proceeding satisfactorily
- If possible, the EmpA will contact the assigned employee(s) (in a manner and at a time that will minimise interruption to their duties) to check:
  - That the job has commenced as planned
  - That they are, indeed, onsite
  - If they have any issues or problems
- The Job Booking Sheet will show that this procedure has been completed.

Outputs

- Job Completion check recorded on the Job Booking Form
• MD-Admin informed of any difficulties, issues, problems, complaints or non-conformances.

**Employee Experience Register**

• Initial data – self reporting
  - To the best of our knowledge, there is no equivalent register available to assess employee experience. It is therefore necessary to establish a baseline by self-reporting. The business recognises that the fidelity of such data is open to question, however it serves as a starting point.
  - Each the business worksite protection employee is to be issued an experience survey (for their duties and the conditions they have worked under) and they are to be encouraged to record their previous experience at certain tasks and work conditions.

• Once the Employee Experience Register Database has been established:
  - Each worksite employee’s database entry will be updated according to the duties and conditions they experience with each assigned work shift.
    - NB: The ongoing experience register for each employee, once a body of ongoing data is accumulated, with have *precedence* over their initially self-reported experience survey data.
  - The experience database should provide information useful when selecting the most suitable employee for any given job.
  - Assignment of work to employees should, so far as is reasonable and does not impede the delivery of BEST SERVICE to clients, allow employees to refresh update and extend their experience.

**Outputs**

• Duty experience survey form.
• Work Condition experience survey form.
• Ongoing experience register database updated.
• Experience reports – as required.

**Exceptions - Occurrence Procedure**

• Advise the MD-Admin
• The MD-Admin appoints an investigator
• An occurrence report cover sheet is initiated and is used as the front-piece for all documents arising from the matter – which will be filed in the Accident/Incident Folder

**Outputs**

• An occurrence report cover sheet
• Entry into the accident/incident folder
• Reports to statutory, regulatory or other bodies, as required by the nature of the occurrence
Corrective Action Report (CAR) Procedure

- All reports, feedback or audits regarding any aspect of the business are to be brought to the attention of the MD-Admin at the earliest opportunity.
- The MD-Admin will detail an Officer to write a CAR which details:
  - Each element or point of the audit/report/complaint/ feedback
  - The circumstances of the particular issue/event etc
  - Identify the relevant regulation/standard/policy etc
  - Any relevant report/response from business employees present at the event/place etc
  - The corrective action to be taken to address the issue/problem etc
    - Be sure to consider how future re-occurrence might be avoided – will a new procedure, policy or SWI be required?
- The MD Admin will sign off each point, and so will any other officer of the business whose direct responsibility it might be to address the matters raised in the audit/report/complaint
  - Eg: the senior safety officer of a worksite

Outputs

- CAR completed – reviewed – signed – and a copy forwarded (post/email/fax) to the requesting/auditing body.
Appendix C: Business Resources

Job Booking Sheet

- This is the central record of any given job.
- Once a job order has been received, a job-booking sheet is initiated, and will record the details of the selected/assigned employee.

Job Scope

- The business will endeavour to obtain as much information about each job as it can. Sometimes a client does not fully grasp the safety requirements of the work they plan to undertake in the rail corridor, and so A B Safety Rail Pty Ltd needs as much information as possible in order to best assess the service best suited to the job.
- The business has a standard Job Scope questionnaire which it can use to prompt a verbal telephone query, or to send to a client to seek written details.
- The job scope details assist in choosing employees who have the appropriate knowledge, skills or who need to extend and enhance their experience base.

Job Boards

- There are two Job Boards, which displays Job Booking Sheets with details of:
  - o current jobs and of
  - o pending jobs.
- These must be consulted to check which employees are NOT currently engaged for the requested period, and will not be fatigued if they were assigned to the requested period as a result of working another job before or after (include consideration of their expected travel time).
  - o Persons who are unavailable will not be considered unless their assignments are changed.
  - o Any change in assignment MUST NOT disadvantage or inconvenience any existing (pending or current) job.

Employee Folder

- This is where individual Employee Files are stored.
- They store details of each employee, including copies of their certificates of competency (“tickets”), Category-1 Medical Clearance, contact and other details.
  - o An ongoing record of the employee’s experience will be stored here as well, once that experience register procedure is finalised and commenced.

Job Letter

- This is, essentially a contract between the business and its employee, setting out the assigned job details, pay rates, allowances and other details.
The Job Scope and any other pertinent details, including a Site Map, are attached to this and a copy is sent to the employee. This letter is attached to the Job Booking Sheet for final filing.

**FAID©**

- FAID© is a commercial software package. The business has a commercial licence and uses this system to assist in tracking the fatigue status of its employees.
- FAID© should be checked to see if it indicates a potential fatigue risk if the candidate performs the job.
  - If the FAID© system indicates a risk, the candidate will be considered UNFIT for duty.

**Employee Quick Reference List**

- This is a wall-chart listing employees, together with their qualification grade, their town/suburb of residence and an indication of the confidence with which they may be placed.
- The Quick Reference Sheet assigns a confidence colour code to each employee.
  - **GREEN** employee is able to be used with reasonable surety.
  - **AMBER** employee is new, untested or may have issues. To be used sparingly, and with supervision (wherever possible).
  - **RED** candidate not to be placed, without direct approval of the Managing Director (Operations), who assumes full and personal responsibility for all such placements.

**Site Map**

- The business endeavours to locate each worksite on published rail network maps and will make every effort to provide employees with a copy of maps of their work area.
  - Where necessary, the business may request maps or similar instructions from the client.
- A copy of the site map, if available, will be attached to the Job Sheet.
Appendix D: Original Business Process Diagrams

The following diagrams had been prepared by the Stakeholder Group (with some assistance from the analyst) as part of their initial in-house attempt to document business processes.

Though they were not prepared strictly in accordance with the ATSA Method, the analyst was able to suggest that they strive to represent the transaction of candidate instruments. Whilst not Activity-centric, they approach being transaction-centric, and as such served as a useful resource in the analysis phase.

Figure D1: The “New Employee” Procedure
Figure D2: The “Select Employee” Procedure
Appendix E: List of Goal Directed Actions

The initially identified Goal driven Actions themselves appear, under somewhat arbitrary sequential numberings, collected under their respective parent Processes. Consistent with this early phase of elicitation and analysis, a deliberately broad summary of Goal, Constraints and/or Conditions are recorded in the second column and a broad description of the Transaction itself (as it occurs under the pre-existing Business process, expressed in terms of Roles and of candidate Instruments) is given in the third column.

<table>
<thead>
<tr>
<th>Description</th>
<th>Goals, Conditions &amp; Constraints</th>
<th>Transaction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(A)</strong> New Employee Application PROCESS</td>
<td>Strive to recruit the best possible personnel able to provide professional specialist services to clients. Establish confidence in new employee capabilities and their responsible attitudes.</td>
<td>The HumA or GenA will contact applicant and confirm postal address details. An Employment Information Pack is prepared and posted out. (contains: Employment Enquiry Letter, Employment Application Form, ASU membership, Superannuation form, Information Sheet)</td>
</tr>
<tr>
<td>1 Employment Information Pack</td>
<td>On initial contact, Employment Information Pack (CI 8) may contain commercial details and must not be sent to an incorrect address</td>
<td>The HumA or GenA will contact applicant and confirm postal address details. An Employment Information Pack is prepared and posted out. (contains: Employment Enquiry Letter, Employment Application Form, ASU membership, Superannuation form, Information Sheet)</td>
</tr>
<tr>
<td>2 Update Employment Enquiries Folder</td>
<td>The Business will retain record of applications even if the applicant is unsuccessful.</td>
<td>The HumA or GenA will record the application in the Application Enquiries Folder (CI 9). Initiate a new Employment Suitability Form (CI 7).</td>
</tr>
<tr>
<td>3 Check tickets</td>
<td>Employee must hold valid and current qualifications including medical clearances.</td>
<td>The HumA or GenA will confirm the applicant holds valid and current tickets and medical clearances for the work they're seeking. Record in Employment Suitability Form (CI 7).</td>
</tr>
<tr>
<td>4 Check applicant owns adequate personal protective equipment (PPE)</td>
<td>Employees are required to own all necessary personal safety equipment.</td>
<td>The HumA or GenA will contact the applicant and ask if they own adequate PPE (unless already specified in the application). Record in Employment Suitability Form (CI 7).</td>
</tr>
<tr>
<td>5 Reference checks</td>
<td>establish (so far as is reasonably possible) that new employee is of reliable character as they'll represent the Business to clients on worksites.</td>
<td>The HumA or GenA will contact the named referees provided by the applicant and record in Employment Suitability Form (CI 7). If referees not provided or available, CSA or EL will undertake an investigation through industry contacts and determine suitability.</td>
</tr>
</tbody>
</table>
### Supervised placement

New employees are given an Amber rating by default, and wherever possible are to be placed in jobs where senior hands may observe and mentor them. Application from anyone initially rateable as Red is rejected.

The CSA or EL may revise employee status (see Process (E)). Status is recorded on the Quick Reference Board (CI 6).

### New Client Establishment Procedure PROCESS

**The Business will strive to expand its customer base. All potential clients are to be treated with courtesy and professionalism.**

#### Commence a new client application form

- **Action**: CLO or GenA initiates a fresh instance of a Client Application Form (CI 1).

#### Request credit references and conduct a credit reference check

- **Action**: CLO or GenA requests details of two or more credit referees from the applicant. These are contacted and standardised questions asked. Any doubts must be referred to the CFO.

#### Establish rates schedule

- **Action**: The CFO negotiates a rates schedule for jobs on a per-client basis. Rate categories, and any special clauses, are recorded in the Client File (CI 2).

#### Create Client File (CI 2)

- **Note**: requires input from Action 9 (above)

#### Create MYOB© (CI 26) entry

- **Action**: The CFO establishes an entry in the stand-alone MYOB© software system.

### General Job Booking PROCESS

**Invoice Number left BLANK until generated by MYOB© in after-job process. Large jobs may constitute multiple jobs (under a single invoice). Employee details added after employee selection process**

#### Contact the client

- **Action**: GenA to give CLO correct Client contact details. Necessary to establish that client agrees with job and pay details.

- **Action**: CLO needs the client contact details. Most jobs are for established clients whose contact details are in the Client File (CI 2).

#### Confirm pay rates

- **Action**: CFO is authorised to negotiate fees charged to the client on a per-job basis.

#### Initiate New Job Booking Sheet

- **Action**: JobA

- **Note**: Job Booking Sheet (CI 18)
<table>
<thead>
<tr>
<th>15</th>
<th>Elicit Job Scope:</th>
<th>CLO to Send client a Job Scope Enquiry Form (CI 19). As necessary, also contact to elicit, clarify or confirm details. CSA or qualified senior employee to review.</th>
<th>CLO sends Job Scope Enquiry Form (CI 19) to elicit details. Record details on Job Scope Report (CI 20) for CSA or qualified senior employee to review. Employee Experience Register (CI 33)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>(D)</strong> Employee Selection PROCESS</td>
<td>Identify suitable employee for each specific job. Must be evaluated for their suitability. Two factors: (1) FITNESS for WORK and (2) Location, proximity to the worksite (to reduce travel and LAHA costs to client). NB: Fitness must always take precedence over location. Those who cannot satisfy ALL fitness requirements cannot be used.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Consult Job Booking Sheet (CI 18) &amp; Job Scope Report (CI 20)</td>
<td>JobA to ascertain location and qualification requirements for employee selection</td>
<td>Job Scope Report (CI 20) Job Booking Sheet (CI 18)</td>
</tr>
<tr>
<td>17</td>
<td>Check Employee Location</td>
<td>EmpA gives preference to employees who live closest to the work. For job requiring multiple employees, ensure that at least one has prior site familiarity wherever possible, and give others opportunity to become familiar</td>
<td>Quick Reference Board (CI 6)</td>
</tr>
<tr>
<td>18</td>
<td>Check Status</td>
<td>EmpA to select candidate employees by Status Colour (Green &gt; Amber &amp; avoid Red)</td>
<td>Quick Reference Board (CI 6)</td>
</tr>
<tr>
<td>19</td>
<td>Check Job Boards (CI 23 &amp; 24)</td>
<td>EmpA ensures candidate is available at the scheduled time for the job and will not become fatigued because of other jobs and/or travel.</td>
<td>Pending Jobs Board (CI 23) Current Jobs Board (CI 24)</td>
</tr>
<tr>
<td>No.</td>
<td>Step Description</td>
<td>Details</td>
<td>Related Documents</td>
</tr>
<tr>
<td>-----</td>
<td>----------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td>20</td>
<td>Select Candidate</td>
<td>Emp A using preceding checks, select best candidate employee(s)</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Confirm Fitness: Check Qualifications</td>
<td>Emp A confirms employee holds all necessary qualification for the duties by the proposed job.</td>
<td>Job Scope Report (CI 20) Employee Folder (CI 5)</td>
</tr>
<tr>
<td>22</td>
<td>Confirm Fitness: Check Experience</td>
<td>Emp A to gauge match between job requirements and employees experience</td>
<td>Job Scope Report (CI 20) Employee Experience Register (CI 33)</td>
</tr>
<tr>
<td>23</td>
<td>Confirm Fitness: Check Cat-1</td>
<td>Emp A to confirm employee’s medical certification is valid and current</td>
<td>Employee Folder (CI 5)</td>
</tr>
<tr>
<td>24</td>
<td>Confirm Fitness: Check Induction</td>
<td>Emp A to confirm employee has undertaken all current inductions</td>
<td>Employee Folder (CI 5)</td>
</tr>
<tr>
<td>25</td>
<td>Confirm Fitness: Check FAID®</td>
<td>FtgA uses 3rd party application used to estimate if job exceeds safe fatigue levels for the employee</td>
<td>FAID® (CI 25)</td>
</tr>
<tr>
<td>26</td>
<td>Consult Employee: Availability</td>
<td>EmpA: Employee to confirm availability for job</td>
<td>EmpA contacts employee (refer Employee File (CI 4)) asks about location and time (refer Job Booking Sheet (CI 18) or Job Scope Report (CI 20)) as per Suitability Checklist (CI 14)</td>
</tr>
<tr>
<td>27</td>
<td>Consult Employee: Pay rate, Travel &amp; LAHA</td>
<td>EmpA: Employee to agree to pay rates, allowances and other conditions in advance</td>
<td>EmpA contacts employee (refer Employee File (CI 4)) asks about pay rate and allowances (refer Job Booking Sheet (CI 18) or Job Scope Report (CI 20)) as per Suitability Checklist (CI 14)</td>
</tr>
<tr>
<td>28</td>
<td>Consult Employee: D&amp;A, Fatigue, PPE</td>
<td>Employee to confirm knowledge of, and compliance with Drug &amp; Alcohol, Fatigue and Personal Protective Equipment policies</td>
<td>EmpA contacts employee (refer Employee File (CI 4)) elicits statement of compliance (refer Job Booking Sheet (CI 18) or Job Scope Report (CI 20)) as per Suitability Checklist (CI 14)</td>
</tr>
<tr>
<td>(E)</td>
<td>Employee Experience Register PROCESS</td>
<td>Track the work experience of employees to allow best selection for client needs and assist employees to lift their skills and qualifications.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Record Location</td>
<td>By location, to give future preference to employees with experience of specific job locations.</td>
<td>Employee Folder (CI 5) Employee Experience Register (CI 33)</td>
</tr>
<tr>
<td>---</td>
<td>-----------------</td>
<td>----------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>30</td>
<td>Record Work Nature</td>
<td>By nature of work, to give future preference to employees with experience of specific job conditions.</td>
<td>Employee Folder (CI 5) Employee Experience Register (CI 33)</td>
</tr>
<tr>
<td>31</td>
<td>Feedback</td>
<td>Elicit feedback from client and/or supervising senior employee</td>
<td>CLO Quick Reference Board (CI 6)</td>
</tr>
<tr>
<td>(F)</td>
<td>General Invoicing PROCESS</td>
<td>Generate accurate and prompt invoices for each service provision and forward these to the client at the earliest opportunity.</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>Confirm details</td>
<td>Contact client and verbally confirm</td>
<td>CLO</td>
</tr>
<tr>
<td>33</td>
<td>Enter into MYOB®</td>
<td>Charge details entered into MYOB® (CI 26).</td>
<td>CFO NB: Only CFO authorised to use</td>
</tr>
</tbody>
</table>
| 34 | Generate invoice number | MYOB® (CI 26) generates a unique invoice number (CI 29). CFO only. | CFO  
Note: for large jobs over multiple sites or times, the MD-Admin may choose to generate a single Invoice Number, which requires delaying this step till all details are confirmed. |
| 35 | Generate invoice & forward to client | CFO only. Print off invoice form from MYOB® (CI 26) | CFO prints invoice (CI 29) from MYOB® (CI 26)  
GenA then posts or faxes to client. |
| 36 | Check for payment | Employee pays are a regular fixed outflow, whilst invoice payments are irregular. Delays in payment to be actioned at the highest level. | GenA checks incoming mail for payments. CFO to check online banking account for direct payments. CFO to authorise the CLO to issue standard reminder letters etc in the event of overdue payment. (Overdue Notice) Escalation cases handled by CFO. |

Table E1: Goal Driven Actions  
(NB: Action numbering approximates temporal sequencing)
Appendices
Appendix F: List of Single Instance Nodes (SINs)

The table identifies candidate Instruments with numbers from Appendix E above. It must be kept in mind that these candidate Instruments are drawn from extant Business documentation and are very likely to be radically altered and expanded during the re-design phase.

Each SIN is given a brief description and some notes about its Goal or the linkages it has to other doings. Side notes and comments are recorded against many SINs indication issues encountered by the Analyst, together with possible ideas for the re-design phase. The Role most closely associated with the SIN is also identified.

<table>
<thead>
<tr>
<th>SIN No.</th>
<th>Description</th>
<th>Role</th>
<th>Candidate Instrument (Table 5.5)</th>
<th>Instrument/Transaction Type</th>
<th>Goal (Links)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Receive employment application</td>
<td>GenA</td>
<td>(CI 3) Employment application</td>
<td>Document</td>
<td>Receive from applicant for employment</td>
</tr>
<tr>
<td>1.2</td>
<td>Record application</td>
<td>HumA</td>
<td>(CI 9) Employment Enquiries</td>
<td>Record</td>
<td>Keep records of all persons seeking employment</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Folder</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td>Confirm address</td>
<td>HumA</td>
<td>(CI 3) Employee application</td>
<td>Record</td>
<td>provide business details only to legitimate applicant</td>
</tr>
<tr>
<td>1.4</td>
<td>Dispatch Info</td>
<td>HumA</td>
<td>(CI 8) Employment Information</td>
<td>Document</td>
<td>Advise potential employee of rights and obligations</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Pack</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td>Initiate new Employment</td>
<td>HumA</td>
<td>(CI 17) Employment Suitability</td>
<td>Record</td>
<td>Follow correct procedure to confirm all necessary details of employees</td>
</tr>
<tr>
<td></td>
<td>Suitability Form</td>
<td></td>
<td>Form</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td>Confirm ticket status</td>
<td>HumA</td>
<td>(CI 17) Employment Suitability</td>
<td>Record</td>
<td>Note if applicant holds tickets that are are current and suitable. IF NOT: terminate process, but retain record. (applicant may re-apply at later date)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Form</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2</td>
<td>Confirm medical status</td>
<td>HumA</td>
<td>(CI 17) Employment Suitability</td>
<td>Record</td>
<td>Note if applicant holds current medical clearance. IF NOT: terminate process, but retain record. (applicant may re-apply at later date)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Form</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 4.1 Confirm PPE HumA (CI 17) Employment Suitability Form
- **Record** Note if applicant understands & owns adequate Personal Protective Equipment. If NOT: HOLD application status open if all other checks clear – DO NOT EMPLOY until passed.

### 5.1 Referee contact info. HumA (CI 17) Employment Suitability Form
- **Read** Extract Referee contact details from Application.

### 5.2 Contact Employee to elicit Referee details HumA (CI 3) Employment application
- **Communiqué** If missing, unclear, expired etc, then elicit referee contact details.
  
  NB: those recently employed by competitor, or with dubious history, may be hesitant to comply.

### 5.3 Contact Referees HumA (CI 17) Employment Suitability Form
- **Record** Follow standard questions. Record responses.

### 5.4 Evaluate Employee Suitability CSA (CI 17) Employment Suitability Form
- **Record** Read & evaluate reports from referees – determine suitability.

### 5.5 Evaluate Employee Suitability CSA (CI 17) Employment Suitability Form
- **Record** The CSA may over-ride referee reports (or the lack of them) based on personal interview, personal knowledge etc.

### 5.6 Decide suitability CSA
- **Communiqué** CSA has final decision on suitability

### 5.7 Record Suitability HumA (CI 17) Employment Suitability Form
- **Record** Apply (default) AMBER status

### 5.8 Advise Applicant GenA (CI 3) Employment application
- **Communiqué** Send advise of successful application to contact address as given in Employment Application

### 6.1 Supervised Placement CSA (CI 17) Employment Suitability Form
- **Record** Decide if the employee can only be placed under supervision till reviewed.
  
  - employee only suitable for multi-person jobs where senior is available

---

310
<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Responsible</th>
<th>Office/Location</th>
<th>Action</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Elicit Client Credit Referee details</td>
<td>CLO</td>
<td>(CI 1)</td>
<td>Record</td>
<td>Locate and record credit referees</td>
</tr>
<tr>
<td>8.2</td>
<td>Contact Credit Referee</td>
<td>CLO</td>
<td>(CI 1)</td>
<td>Record</td>
<td>Evaluate client’s reliability and promptness with payments.</td>
</tr>
<tr>
<td>8.3</td>
<td>Evaluate Client Suitability</td>
<td>CFO</td>
<td>(CI 1)</td>
<td>Record</td>
<td>The CFO may over-ride referee reports</td>
</tr>
<tr>
<td>8.4</td>
<td>Accept Client</td>
<td>CFO</td>
<td>(CI 1)</td>
<td>Record</td>
<td>CFO has final decision on suitability</td>
</tr>
<tr>
<td>9.1</td>
<td>Charge Rate</td>
<td>CFO</td>
<td>(CI 15)</td>
<td>Read/Select</td>
<td>the CFO negotiates and selects a charge rate for the Client from the rates schedule</td>
</tr>
</tbody>
</table>
| 9.2 | Record Rates                                                               | CLO         | (CI 2)          | Record | At instruction of the CFO, the CLO records the rates category (and any special conditions or clauses) in the Client File.  
NOTE: infers the prior existence of pre-defined categories to select from & some comment field for additional clauses ... also infers existence of some process(es) whereby these categories and rates schedules are designed/defined. |
<p>| 10.1 | Initiate new Client File                                                   | CLO         | (CI 2)          | Record | Note: this is out of sequence, cursory &amp; predates SIN 9.2. It’s a direct consequence of SIN 8.4 |
| 11.1 | Create MYOB® Entry                                                         | CFO         | (CI 26)         | Record | CFO creates a MYOB® account entry for the new client. |
| 12.1 | Confirm Contact details                                                    | GenA        | (CI 2)          | Read   | Possibly unnecessary to record here. |
| 12.2 | Give Contact details to CLO                                               | GenA        | Communiqué      |        | Possibly unnecessary to record here. |
| 13.1 | Solicit jobs                                                               | GenA        | (CI 2)          | Communiqué (reports to the MO) | The GenA performs a weekly “ring around” of Clients seeking jobs. |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>13.2</td>
<td>Receive Job Enquiry</td>
<td>GenA</td>
<td>Communiqué</td>
<td>GenA receives call, email, fax etc requesting Business service for a job.</td>
</tr>
<tr>
<td>13.3</td>
<td>Initiate Job Booking Sheet</td>
<td>GenA</td>
<td>Write</td>
<td>Record details of Job request from Client</td>
</tr>
<tr>
<td>13.4</td>
<td>Advise JobA</td>
<td>GenA</td>
<td>Communiqué</td>
<td>Promptly send Job Booking Sheet to JobA</td>
</tr>
</tbody>
</table>
| 13.5 | Confirm job rates with Client | CLO | Communiqué | After selecting an employee, pay rates and allowances, confirm Client agrees to rates for the job. 
*Note: Should possibly occur in parallel to employee agreeing to rates (27)* |
| 13.6 | Record pay & Bonuses | JobA | Write | Record all relevant financial details, including employee payments |
| 14.1 | Receive Job Booking Sheet | JobA | Read | Receive from GenA 
*NOTE: perhaps use a status flag to indicates if a Job is active: either change status or by internal transaction convert to a new instrument (job note to job sheet)* |
| 15.1 | Send Job Scope Enquiry Form to client | CLO | Communiqué | Provides a mechanism to elicit important details from client |
| 15.2 | Exact Location | JobA | Record | Vital data for selecting and, later, briefing employee(s) |
| 15.3 | Job Duration | JobA | Record | Vital data for selecting and, later, briefing employee(s) 
*May impact on costs* |
<p>| 15.4 | Start-time, incl PreWork Briefing | JobA | Record | Vital data for selecting and, later, briefing employee(s) |
| 15.4 | Expected End Time | JobA | Record | Vital data for selecting and, later, briefing employee(s) |
| 15.5 | Number of persons on work site | JobA | Record | Vital data for selecting and, later, briefing employee(s) |
| 15.6 | Physical extent of worksite | JobA | Record | Vital data for selecting and, later, briefing employee(s) |</p>
<table>
<thead>
<tr>
<th>15.7</th>
<th>Physical extent of worksite</th>
<th>JobA</th>
<th>(CI 20) Job Scope Report</th>
<th>Record</th>
<th>Vital data for selecting and, later, briefing employee(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.8</td>
<td>Transport and Comms requirements</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Record</td>
<td>Vital data for selecting and, later, briefing employee(s)</td>
</tr>
<tr>
<td>15.9</td>
<td>Nature of work</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Record</td>
<td>Vital data for selecting and, later, briefing employee(s)</td>
</tr>
</tbody>
</table>
| 15.10 | Expert Review of Scope | CSA | (CI 20) Job Scope Report | Flag | The CSA should review job scope details and assess for completeness or expected issues. 
\*NOTE: Job location and time may be sufficient to initiate the selection of employees, but that process cannot be completed until the job scope details are available for comparison against employee experience, skills and qualification. |
| 16.1  | Ascertain location | EmpA | (CI 20) Job Scope Report or (CI 18) Job Booking Sheet | Read | Identify Location of Job – used to select Employees (preference given to employees close to job to minimise travel & other bonus costs for the client) 
\*NOTE: an employee may not be home because they're currently placed elsewhere (trade off decisions: Travel vs LAHA costs) 
\*NOTE: implies the need to keep a live “current location” field on the Quick Reference Sheet |
### 16.2 Ascertain qualification requirements

<table>
<thead>
<tr>
<th>EmpA</th>
<th>(CI 20) Job Scope Report</th>
<th>Read</th>
</tr>
</thead>
</table>
| Identify nature of the job – used to select Employees. Suitable and current qualifications are an ABSOLUTE pre-condition. Pertinent experience is preferred. Opportunity in (multi-person jobs) to allow employees to enhance their experience profile (under guidance from experienced seniors) is encouraged.  
NOTE: perhaps a series of Flags to indicate multi-person jobs.  
NOTE: in the experience register – there may be need to identify key TYPES of work site and work such that experience can be easily tracked – and opportunities to acquire experience taken into account. |

### 17.1 Check Employee Location

<table>
<thead>
<tr>
<th>EmpA</th>
<th>(CI 6) Quick Reference Board</th>
<th>Read</th>
</tr>
</thead>
</table>
| Preference given to employees located near job, to minimise travel & LAHA costs to the client.  
NOTE: Implies usefulness of some kind of Travel Cost & LAHA Cost “ready reckoner” to assist in selection of Employee (by location) to minimise Client costs.  
Travel bonus is calculated as a function of distance – infer use for an automated distance calculator  
NOTE: as for 16.1, implies the need to keep a “live” current location field on the Quick Reference Board |
| 17.2 | Give selection preference by site familiarity | EmpA | (CI 14) Employee File | Read | For job requiring multiple employees, ensure that at least one has prior site familiarity wherever possible, and give others opportunity to become familiar. NB: Clients request close & experienced workers; the Business wants employees to build experience & employees want to increase opportunities for placement. NOTE: implies usefulness of noting which employees can act (and have acted) as 'mentors' – recorded on employee file but somehow available in Quick Reference Board. NOTE: implies the need for a LISTING of job types and conditions – used in the Experience Register, updated by job placement & review, reflected in the Quick Reference Board. NOTE: implies some usefulness in tracking up-to-date experience (for specific job types & conditions).

| 18.1 | Check Employee Status | EmpA | (CI 6) Quick Reference Board | Read | Green employees are clear – Amber need mentoring &/or assessment – Red to be avoided at all costs. NOTE: 3-state grading is arbitrary and imprecise. A computerised system could enhance under more than one dimension (eg: Location (auto-calculated in real time from Job location: Mentoring capacity: nature of Job (from experience register & Job Scope)).

| 19.1 | Check Current Job Board | EmpA | (CI 24) Current Job Board | Read | Employees currently on placement are unavailable.

<p>| 19.2 | Check Pending Job Board | EmpA | (CI 23) Pending Job Board | Read | Employees booked for jobs which clash are unavailable. |</p>
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Related Documents</th>
<th>Action</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>19.3</td>
<td>Over-ride placement &amp; re-allocate (i)</td>
<td>CSA (CI 18) Job Booking Sheet</td>
<td>Write (ie: change, modify)</td>
<td>CSA can re-allocate employees service as many jobs as possible. This is an “exception case” – sufficiently rare to not require implementation in the system per se, but “editing” of prior decisions must be possible. NOTE: Implies possible need for consistency checking functionality to ensure all implicit effects of changes are actioned correctly such that no employee, client or job is put into an error state. NOTE: this is a non-trivial computational task – possibly beyond the means of the current project.</td>
</tr>
<tr>
<td>19.4</td>
<td>Over-ride placement &amp; re-allocate (ii)</td>
<td>CSA (CI 22) Job Sheet</td>
<td>Write (changes to (CI 18) Job Booking Sheet, infer changes to (CI 22) Job Sheet)</td>
<td>CSA has the authority to re-allocate employees to maximise the ability to service as many jobs as possible.</td>
</tr>
<tr>
<td>19.5</td>
<td>Over-ride placement &amp; re-allocate (iii)</td>
<td>CSA (CI 24) Current Job Board</td>
<td>Write (ie: change, modify, move instance(s) of (CI 22) Job Sheet)</td>
<td>CSA has the authority to re-allocate employees to maximise the ability to service as many jobs as possible.</td>
</tr>
<tr>
<td>19.6</td>
<td>Over-ride placement &amp; re-allocate (iv)</td>
<td>CSA (CI 23) Pending Job Board</td>
<td>Write (ie: change, modify, move (CI 22) Job Sheet)</td>
<td>CSA has the authority to re-allocate employees to maximise the ability to service as many jobs as possible.</td>
</tr>
<tr>
<td>20.1</td>
<td>Select candidate</td>
<td>EmpA Unspecified candidate Instrument some temporary listing of candidate employee(s)</td>
<td>Cache</td>
<td>Following preceding checks, a candidate employee is selected for further checks. Possibly not a “doing” but a consequence. NOTE: suggests a Flag – perhaps flag Employee or append employeeID to Job Sheet – pending confirmation of selection</td>
</tr>
</tbody>
</table>
| 21.1 | Check Qual’s | EmpA | (CI 20) Job Scope Report | Read | Ascertain Qualifications required for the job
NOTE: suggests usefulness of a listing of qualifications vs: job types
NOTE: some clients specify the qualifications needed, but these still need to be confirmed by the Business |
| 21.2 | Confirm Qual’s | EmpA | (CI 14) Employee File | Read | Confirm candidate Employee holds appropriate current Qualifications |
| 21.3 | Pass Qual’s Check | EmpA | (CI 14) Suitability checklist | Write | Record pass |
| 21.4 | Fail Qual’s Check | EmpA | temporary listing of candidate employee(s) | Cache | if check failed, delete temporary listing of candidate employee(s)
Return to SIN 20.1 & select new candidate. If none available, client’s job may have to be refused. |
| 22.1 | Check Experience | EmpA | (CI 20) Job Scope Report | Read | Ascertain Experience category required for the job
NOTE: implies the need for a LISTING of job types and conditions – used in the Experience Register, updated by job placement & review, reflected in the Quick Reference Board |
| 22.2 | Confirm Experience | EmpA | (CI 14) Employee File | Read | Confirm candidate Employee has appropriate experience
NOTE: implies some usefulness in tracking up-to-date experience (for specific job types & conditions) |
| 22.3 | Pass Experience Check | EmpA | (CI 14) Suitability checklist | Write | If a multi-employee job, might act as mentor to other less experienced employee(s)
NOTE: if selected then should update experience register (if job completed satisfactorily) – may impact on Quick Reference ‘colour’ status |
| 22.4 | Fail Experience Check | EmpA | temporary listing of candidate employee(s) | Cache | If a multi-employee job, might be permitted if mentored by more experienced employee.  
*NOTE: if selected then should update experience register (if job completed satisfactorily) – may impact on Quick Reference ‘colour’ status*  
ELSE if check failed, delete temporary listing of candidate employee(s)  
Return to SIN 20.1 & select new candidate. If none available, client’s job may have to be refused. |
<p>| 23.1 | Confirm Medical Cat-1 status | EmpA | (CI 14) Employee File | Read | A simple gateway, irrespective of Job Scope. Employee must hold current medical clearance as issued by a government authorised medical testing agency |
| 23.2 | Fail Medical Cat-1 status: Reject Candidate | EmpA | temporary listing of candidate employee(s) | Cache | Cat-1 not current, cannot use that employee. Return to SIN 20.1 &amp; select new candidate. If none available, client’s job may have to be refused. |
| 23.2 | Fail Medical Cat-1 status: Revise status | EmpA | (CI 6) Quick Reference Board | Write | employee goes to immediate RED status |
| 23.3 | Fail Medical Cat-1 status: Flag for review | EmpA | | Communiqué | Employee file brought to attention of CSA, CFO &amp; ELO |
| 24.1 | Confirm Induction | EmpA | (CI 14) Employee File | Read | A simple gateway, irrespective of Job Scope. Employee must have passed through basic Employee Induction procedure. |
| 24.2 | Fail Induction: Induct | CSA | (CI 42) Employee Induction Package | Physical presence at Business premises required. | If time permits, Employee may be inducted prior to commencement of duties. Requires personal &amp; manual induction procedure. |
| 24.3 | Fail Induction: Reject Candidate | EmpA | temporary listing of candidate employee(s) | Cache Write (delete) | If Induction not recorded, employee not to be used. Return to SIN 20.1 &amp; select new candidate. If none available, client’s job may have to be refused. |
| 25.1 | Estimate job duration | FtgA | (CI 20) Job Scope Report | Read (NB), only the FtgA has had the specific training to use FAID® and holds logon access | Obtain the work hours and days for the job, inclusive of estimated Travel Time. NB: Travel Time is a function of distance. Travel time is a factor in Fatigue calculations. NOTE: A Travel Time calculator, based on the job location and the employees expected start location (usually home) would be useful. |
| 25.2 | Check FAID® | FtgA | (CI 25) FAID® Offline process | 3rd party FAID® estimates the level of fatigue. Employee may not work under risky levels of fatigue – may need to be relieved or refused placement if at risk. (NOTE: system could format data to enhance using FAID®. Future versions of FAID® may allow direct data exchange, effectively linking into the system as a whole.) |
| 25.3 | Advise JobA of FAID® status | FtgA | (CI 25) FAID® Communiqué | Advise the JobA of the result of the FAID® check |
| 25.3 | Confirm FAID® | EmpA | (CI 14) Suitability checklist | Write | A simple gateway, irrespective of Job Scope. Employee must not work if effected by fatigue. |
| 25.4 | Fail FAID® | EmpA | temporary listing of candidate employee(s) | Cache Write (delete) | If check failed Return to SIN 20.1 &amp; select new candidate. Maybe possible to relieve the employee, using extra employee(s), but client may refuse to pay added costs. If no replacement available, and/or client refuses to job-share, the client’s job may have to be refused. |</p>
<table>
<thead>
<tr>
<th>Appendix</th>
<th>Task</th>
<th>Employee Responsibility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>26.1</td>
<td>Read Duration</td>
<td>EmpA</td>
<td>(CI 20) Job Scope Report</td>
</tr>
<tr>
<td>26.1</td>
<td>Check Availability</td>
<td>EmpA</td>
<td>(CI 14) Employee File</td>
</tr>
<tr>
<td>26.2</td>
<td>Confirm availability</td>
<td>EmpA</td>
<td>(CI 14) Suitability checklist</td>
</tr>
<tr>
<td>26.3</td>
<td>Fail Availability.</td>
<td>EmpA</td>
<td>temporary listing of candidate employee(s)</td>
</tr>
<tr>
<td>27.1</td>
<td>Read pay rate, travel &amp; LAHA</td>
<td>EmpA</td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td>27.2</td>
<td>Read Employee Contact details</td>
<td>EmpA</td>
<td>(CI 14) Employee File</td>
</tr>
<tr>
<td>27.3</td>
<td>Check with employee</td>
<td>EmpA</td>
<td>Telephone</td>
</tr>
<tr>
<td>27.4</td>
<td>Confirm pay rates OK</td>
<td>EmpA</td>
<td>(CI 14) Suitability checklist</td>
</tr>
<tr>
<td>27.4</td>
<td>Fail pay rates</td>
<td>EmpA</td>
<td>temporary listing of candidate employee(s)</td>
</tr>
<tr>
<td>28.1</td>
<td>Read Employee Contact details</td>
<td>EmpA</td>
<td>(CI 14) Employee File</td>
</tr>
<tr>
<td>28.2</td>
<td>Check D&amp;A, Fatigue, PPE</td>
<td>EmpA</td>
<td>Telephone</td>
</tr>
<tr>
<td>28.3</td>
<td>Confirm D&amp;A, Fatigue, PPE</td>
<td>EmpA</td>
<td>(CI 14) Suitability checklist</td>
</tr>
<tr>
<td>28.4</td>
<td>Fail D&amp;A, Fatigue, PPE</td>
<td>EmpA</td>
<td>temporary listing of candidate employee(s)</td>
</tr>
<tr>
<td>28.5</td>
<td>Generate (CI 21) Job Letter</td>
<td>JobA</td>
<td>(CI 21) Job Letter</td>
</tr>
<tr>
<td>28.6</td>
<td>Send Job Letter</td>
<td>GenA</td>
<td>(CI 21) Job Letter</td>
</tr>
<tr>
<td>29.1</td>
<td>Record Location</td>
<td>EmpA</td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td>29.2</td>
<td>Record Location</td>
<td>EmpA</td>
<td>(CI 33) Employee Experience Register</td>
</tr>
<tr>
<td>30.1</td>
<td>Record Work Nature</td>
<td>EmpA</td>
<td>(CI 20) Job Scope Report</td>
</tr>
<tr>
<td>30.2</td>
<td>Record Work Nature</td>
<td>EmpA</td>
<td>(CI 33) Employee Experience Register</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>31.1</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 22) Job Sheet</td>
</tr>
<tr>
<td>31.2</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 2) Client File</td>
</tr>
<tr>
<td>31.3</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 34) Client review</td>
</tr>
<tr>
<td>31.3</td>
<td>Record feedback on employee</td>
<td>CLO</td>
<td>(CI 4) Employee File</td>
</tr>
<tr>
<td>31.4</td>
<td>Contact Senior Employee</td>
<td>EmpA</td>
<td>(CI 22) Job Sheet</td>
</tr>
<tr>
<td>31.5</td>
<td>Contact Senior Employee</td>
<td>EmpA</td>
<td>(CI 4) Employee File</td>
</tr>
<tr>
<td>31.6</td>
<td>Contact Senior employee</td>
<td>EmpA</td>
<td>Telephone</td>
</tr>
<tr>
<td>31.7</td>
<td>Record Feedback on employee</td>
<td>EmpA</td>
<td>(CI 4) Employee File</td>
</tr>
<tr>
<td>31.8</td>
<td>Review Status</td>
<td>CSA</td>
<td>(CI 4) Employee File</td>
</tr>
<tr>
<td>31.9</td>
<td>Update Status</td>
<td>EmpA</td>
<td>(CI 6) Quick Reference Board</td>
</tr>
<tr>
<td>32.1</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 22) Job Sheet</td>
</tr>
<tr>
<td>32.2</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 2) Client File</td>
</tr>
</tbody>
</table>
| 32.3 | Contact Client | CLO | (CI 34) Client review | Communiqué | Confirm satisfactory completion of job and adjust charges for any variations in hours, duration etc.  
**NOTE: re-contacting client seems irrelevant, or possibly redundant after SIN 31.3** |
<p>| 33.1 | Obtain Job details | CFO | (CI 18) Job Booking Sheet | Read | Extract fees and charges |
| 33.2 | Enter Charge details into MYOB® | CFO | (CI 26) MYOB® 3rd party software | Write | CFO enters charge details against the Clients’ MYOB® entry |</p>
<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
<th>Responsible</th>
<th>Source</th>
<th>Details</th>
</tr>
</thead>
</table>
| 34.1 | Generate invoice number | CFO | (CI 26) MYOB® | MYOB® (CI 26) generates an invoice number (CI 28)  
**NOTE:** as the invoice number sometimes served as a unique key field “Job Number”, but at other times was not generated until after the job was completed. |
| 34.2 | Advise JobA | CFO | (CI 28) Invoice No | Communiqué  
CFO relays the invoice number to the JobA |
| 34.3 | Record Invoice number | JobA | Job Sheet (CI 22) | **Note:** for large jobs over multiple sites or times, the MD-Admin may choose to generate a single Invoice Number, which requires delaying this step till all details are confirmed. |
| 34.4 | Record Invoice number | JobA | (CI 18) Job Booking Sheet | **Write**  
The Invoice number is recorded on the Job Booking Sheet prior to archiving |
| 35.1 | Generate & Print hardcopy invoice | CFO | (CI 26) MYOB® 3rd party software | Output, (CI 29) Invoice  
The CFO prints off a copy of the invoice |
| 35.2 | Dispatch Invoice | GenA | (CI 29) Invoice | Physical Dispatch  
The AA dispatches the Invoice with the daily Mail  
**NOTE:** the client postal address is included on the invoice by the MYOB® software.  
**NOTE:** production of the envelope etc are trivial “offline” tasks not recorded in this analysis. |
| 36.1 | Check for Payment | GenA | Incoming Mail | **Read**  
The AA checks for inbound payments from clients |
| 36.2 | Advise CFO | GenA | Incoming Mail | Communiqué  
The AA advises and forwards all payments to the CFO |
| 36.3 | Record payment | CFO | (CI 26) MYOB® 3rd party software | **Write**  
Payment instance recorded |
| 36.4 | Process payment | CFO | (CI 31) Online banking | **Write**  
The CFO may process payment via online banking (if appropriate)  
ELSE Manually during weekly banking activity |

**Table F1: SINs**
Appendices
### Appendix G: Activity Conflation Worksheet

The following sheet was assembled in a spreadsheet application. It was created by listing all identified SINs and sorting them according to their Role. The list was then manually sorted and grouped by the Analyst’s understanding of their Goal and constraints.

Processes, as described by the Business, are not strictly an element of the ATSA Method and the Analyst is not constrained by them. They may however, serve as a useful means of simplifying large blocks of data. The sort by Goal in this case produced a somewhat natural clustering of eight processes, the last two of which were not included in the final redesign (as discussed in chapter 5).

As might be expected by the observed self-completing nature of the Combined Activity Table (CAT), it became apparent that a number of small SIN-like doings were required for completeness. These have been added to this worksheet during the Conflation of Activities and the construction of the CAT. They are identifiable here by the absence of a SIN number.

<table>
<thead>
<tr>
<th>Process</th>
<th>ACTY #</th>
<th>SIN #</th>
<th>Description</th>
<th>Role</th>
<th>Candidate Instrument</th>
<th>Instrument/Transaction Type</th>
<th>Goal (Links)</th>
</tr>
</thead>
<tbody>
<tr>
<td>?? Commence new client application form</td>
<td>??</td>
<td>CLO</td>
<td>(CI 1) Client Application Form</td>
<td>Write</td>
<td>CLO commences a new Client Application Form and forwards to CFO. NOTE: unlike new employees (who are asked to fill in forms themselves) new client details are elicited</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.1 Elicit Client Credit Referee details</td>
<td>8.1</td>
<td>CLO</td>
<td>(CI 1) Client Application Form</td>
<td>Record</td>
<td>Locate and record credit referees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.2 Contact Credit Referee</td>
<td>8.2</td>
<td>CLO</td>
<td>(CI 1) Client Application Form</td>
<td>Record</td>
<td>Evaluate client’s reliability and promptness with payments.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.3 Evaluate Client Suitability</td>
<td>8.3</td>
<td>CFO</td>
<td>(CI 1) Client Application Form</td>
<td>Record</td>
<td>The CFO may over-ride referee reports</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.4 Accept Client</td>
<td>8.4</td>
<td>CFO</td>
<td>(CI 1) Client Application Form</td>
<td>Record</td>
<td>CFO has final decision on suitability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.1 Charge Rate</td>
<td>9.1</td>
<td>CFO</td>
<td>(CI 15) Rates Schedule</td>
<td>Read/Select</td>
<td>the CFO negotiates and selects a charge rate for the Client from the rates schedule</td>
<td></td>
<td></td>
</tr>
<tr>
<td>??? advice CLO =&gt;ACTY4</td>
<td>???</td>
<td>CFO</td>
<td>Communiqué</td>
<td>Advise CLO of Rate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>??? advice CLO =&gt;ACTY4</td>
<td>???</td>
<td>CFO</td>
<td>Communiqué</td>
<td>instruct CLO to commence new (CI 2) Client file</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11.1 Create MYOB® Entry</td>
<td>11.1</td>
<td>CFO</td>
<td>(CI 26) MYOB® 3rd party software</td>
<td>Record</td>
<td>CFO creates a MYOB® account entry for the new client.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 10.1 Initiate new Client File

**CLO** (CI 2) **Client File**

**Record**

Note: this is out of sequence, cursory & predates SIN 9.2. It’s a direct consequence of SIN 8.4

### 9.2 Record Rates

**CLO** (CI 2) **Client File**

**Record**

At instruction of the CFO, the CLO records the rates category (and any special conditions or clauses) in the Client File. **NOTE:** infers the prior existence of pre-defined categories to select from & some comment field for additional clauses … also infers existence of some process(es) whereby these categories and rates schedules are designed/defined.

<table>
<thead>
<tr>
<th>1.1</th>
<th>Receive employment application</th>
<th>Hum A</th>
<th>(CI 3) Employment Application</th>
<th>Document</th>
<th>receive from applicant for employment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>Record application</td>
<td>Hum A</td>
<td>(CI 9) Employment Enquiries Folder</td>
<td>Record</td>
<td>Keep records of all persons seeking employment</td>
</tr>
<tr>
<td>1.3</td>
<td>Confirm address</td>
<td>Hum A</td>
<td>(CI 3) Employee Application</td>
<td>Record</td>
<td>provide business details only to legitimate applicant</td>
</tr>
<tr>
<td>1.4</td>
<td>Dispatch Info</td>
<td>Hum A</td>
<td>(CI 8) Employment Information Pack</td>
<td>Document</td>
<td>Advise potential employee of rights and obligations</td>
</tr>
<tr>
<td>2.1</td>
<td>Initiate new Employment Suitability Form</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Record</td>
<td>Follow correct procedure to confirm all necessary details of employees</td>
</tr>
<tr>
<td>5.1</td>
<td>Referee contact info.</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Read</td>
<td>Extract Referee contact details from Application.</td>
</tr>
<tr>
<td>5.2</td>
<td>Contact Employee to elicit Referee details</td>
<td>Hum A</td>
<td>(CI 3) Employment application</td>
<td>Communiqué</td>
<td>If missing, unclear, expired etc, then elicit referee contact details. NB: those recently employed by competitor, or with dubious history, may be hesitant to comply.</td>
</tr>
<tr>
<td>5.3</td>
<td>Contact Referees</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Record</td>
<td>Follow standard questions. Record responses.</td>
</tr>
<tr>
<td>3.1</td>
<td>Confirm ticket status</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Record</td>
<td>Note if applicant holds tickets that are current and suitable. IF NOT: terminate process, but retain record. (applicant may re-apply at later date)</td>
</tr>
<tr>
<td>3.2</td>
<td>Confirm medical status</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Record</td>
<td>Note if applicant holds current medical clearance. IF NOT: terminate process, but retain record. (applicant may re-apply at later date)</td>
</tr>
<tr>
<td>4.1</td>
<td>Confirm PPE</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Record</td>
<td>Note if applicant understands &amp; owns adequate Personal Protective Equipment. IF NOT: HOLD application status open if all other checks clear – DO NOT EMPLOY until passed.</td>
</tr>
<tr>
<td>Step</td>
<td>Task</td>
<td>Responsible</td>
<td>Document</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>-------------</td>
<td>----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>advise CSA</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Communiqué Advise CSA of new potential employee &amp; details</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>receive approval request</td>
<td>CSA</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Read receive (CI 17) Employment Suitability Form from HumA</td>
<td></td>
</tr>
<tr>
<td>5.4</td>
<td>Evaluate Employee Suitability</td>
<td>CSA</td>
<td>(CI 17) Employment Suitability Form</td>
<td>record Read &amp; evaluate reports from referees – determine suitability.</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Supervised Placement</td>
<td>CSA</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Record The CSA may over-ride referee reports (or the lack of them) based on personal interview, personal knowledge etc.</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>decide suitability</td>
<td>CSA</td>
<td></td>
<td>Communiqué CSA has final decision on suitability</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>receive CSA decision</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Read receive CSA decision on appointment &amp; status</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>Initiate Employee File</td>
<td>Hum A</td>
<td>(CI 14) Employee File</td>
<td>Write initiate new specific instance of employee file</td>
<td></td>
</tr>
<tr>
<td>5.7</td>
<td>Record Suitability</td>
<td>Hum A</td>
<td>(CI 17) Employment Suitability Form</td>
<td>Write Apply (default) AMBER status</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>add to quick reference board</td>
<td>Hum A</td>
<td>(CI 6) Quick Reference Board</td>
<td>Write add new entry to Quick Reference Board (name, location, quals and colour status)</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>Advise Applicant</td>
<td>Hum A</td>
<td>(CI 3) Employment application</td>
<td>Communiqué Send advise of successful application to contact address as given in Employment Application</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>advise FtgA</td>
<td>Hum A</td>
<td>(CI 14) Employee File</td>
<td>Communiqué advise FtgA of new employee so FAID® entry can occur</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>Advise PM</td>
<td>Hum A</td>
<td>(CI 14) Employee File</td>
<td>Communiqué advise PM of new employee so financial system entry can occur</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>receive advise of new employee</td>
<td>PM</td>
<td>(CI 14) Employee File</td>
<td>Read access details of new employee</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>establish pay entry Online Banking</td>
<td>PM</td>
<td>(CI 30) Online Banking</td>
<td>Write PM creates a new account</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>establish pay entry MYOB®</td>
<td>PM</td>
<td>(CI 26) MYOB® 3rd party software</td>
<td>Write PM creates a new wages entry in the MYOB® system for the new employee’s pay.</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>receive advise of new employee</td>
<td>FtgA</td>
<td>(CI 14) Employee File</td>
<td>Read access details of new employee</td>
<td></td>
</tr>
<tr>
<td>Step</td>
<td>Description</td>
<td>Responsible Party</td>
<td>Additional Information</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-------------------</td>
<td>------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13.1</td>
<td>Solicit jobs</td>
<td>Gen A</td>
<td>(CI 2) Client File</td>
<td>The GenA performs a weekly “ring around” of Clients seeking jobs. (under MO)</td>
<td></td>
</tr>
<tr>
<td>13.2</td>
<td>Receive Job Enquiry</td>
<td>Gen A</td>
<td>(CI 18) Job Booking Sheet</td>
<td>GenA receives call, email, fax etc requesting Business service for a job.</td>
<td></td>
</tr>
<tr>
<td>13.3</td>
<td>Initiate Job booking Sheet</td>
<td>Gen A</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Write record details on a new instance Job Booking Sheet</td>
<td></td>
</tr>
<tr>
<td>13.4</td>
<td>Advise JobA and CLO</td>
<td>Gen A</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Communique forward Job Booking Sheet to JobA and CLO</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>Receive notification from GenA</td>
<td>CLO</td>
<td>(CI 18) Job Booking Sheet</td>
<td>CLO receives notification of Job request from GenA</td>
<td></td>
</tr>
<tr>
<td>15.1</td>
<td>Send Job Scope Enquiry Form to client</td>
<td>CLO</td>
<td>(CI 19) Job Scope Enquiry Form</td>
<td>Provides a mechanism to elicit important details from client</td>
<td></td>
</tr>
<tr>
<td>14.1</td>
<td>Receive Job Booking Sheet</td>
<td>JobA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Read receive from GenA NOTE: perhaps use status flag to indicates if a Job is active: either change status or by internal transaction convert to a new instrument (job note to job sheet)</td>
<td></td>
</tr>
<tr>
<td>15.2</td>
<td>Exact Location</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write Vital data for selecting and, later, briefing employee(s)</td>
<td></td>
</tr>
<tr>
<td>15.3</td>
<td>Job Duration</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write Vital data for selecting and, later, briefing employee(s) May impact on costs</td>
<td></td>
</tr>
<tr>
<td>15.4</td>
<td>Start-time, incl PreWork Briefing</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write Vital data for selecting and, later, briefing employee(s)</td>
<td></td>
</tr>
<tr>
<td>15.5</td>
<td>Expected End Time</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write Vital data for selecting and, later, briefing employee(s)</td>
<td></td>
</tr>
<tr>
<td>15.6</td>
<td>Number of persons on work site</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write Vital data for selecting and, later, briefing employee(s)</td>
<td></td>
</tr>
<tr>
<td>15.7</td>
<td>Physical extent of worksite</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write Vital data for selecting and, later, briefing employee(s)</td>
<td></td>
</tr>
<tr>
<td>15.9</td>
<td>Nature of work</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write</td>
<td>Vital data for selecting and, later, briefing employee(s)</td>
</tr>
<tr>
<td>15.8</td>
<td>Transport and communications requirements</td>
<td>JobA</td>
<td>(CI 20) Job Scope Report</td>
<td>Write</td>
<td>Vital data for selecting and, later, briefing employee(s)</td>
</tr>
<tr>
<td>??</td>
<td>Combine Job Scope with Job Booking</td>
<td>JobA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Write (Append)</td>
<td>Join Job Scope Report to Job Booking Sheet</td>
</tr>
<tr>
<td>??</td>
<td>advise EmpA</td>
<td>JobA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Communiqué</td>
<td>advise EmpA to initiate employee selection</td>
</tr>
<tr>
<td>??</td>
<td>advise CFO</td>
<td>JobA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Communiqué</td>
<td>advise CFO to initiate financial system entry</td>
</tr>
<tr>
<td>??</td>
<td>receive details</td>
<td>CFO</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Read</td>
<td>receive Job Booking Sheet from JobA</td>
</tr>
<tr>
<td>33.2</td>
<td>Enter Charge details into MYOB®</td>
<td>CFO</td>
<td>(CI 26) MYOB® 3rd party software</td>
<td>Write</td>
<td>CFO enters charge details against the Clients’ MYOB® entry. NOTE: the MD-Admin was very reticent to reveal the full details of financial doings</td>
</tr>
<tr>
<td>34.1</td>
<td>Generate invoice number (CI 28)</td>
<td>CFO</td>
<td>(CI 26) MYOB® 3rd party software</td>
<td>Read</td>
<td>MYOB® (CI 26) generates an invoice number (CI 28) NOTE: the Analyst observed some inconsistency here, as the invoice number sometimes served as a unique key field “Job Number”, but at other times was not generated until after the job was completed.</td>
</tr>
<tr>
<td>34.2</td>
<td>Advise JobA</td>
<td>CFO</td>
<td>(CI 28) Invoice Number</td>
<td>Communiqué</td>
<td>CFO relays the invoice number to the JobA</td>
</tr>
<tr>
<td>??</td>
<td>receive Invoice Number</td>
<td>JobA</td>
<td>(CI 28) Invoice Number</td>
<td>Read</td>
<td>receive from CFO</td>
</tr>
<tr>
<td>??</td>
<td>Initiate Job Sheet</td>
<td>JobA</td>
<td>(CI 22) Job Sheet</td>
<td>Write</td>
<td>Note: Job Sheet is a kind of “token” version of the Job Booking Sheet – which is placed on the Job Boards</td>
</tr>
<tr>
<td>34.3</td>
<td>Record Invoice number (CI 28)</td>
<td>JobA</td>
<td>(CI 22) Job Sheet</td>
<td>Write</td>
<td>Note: for large jobs over multiple sites or times, the MD-Admin may choose to generate a single Invoice Number, which requires delaying this step till all details are confirmed.</td>
</tr>
<tr>
<td>13.6</td>
<td>Record pay rate</td>
<td>JobA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Write</td>
<td>Record all relevant financial details, including employee payments. Charge rates are recorded on Client file. Employee payments are standardised.</td>
</tr>
<tr>
<td>??</td>
<td>post Job Sheet to Jobs Pending Board</td>
<td>JobA</td>
<td>(CI 23) Pending Jobs Board</td>
<td>Write</td>
<td>pin the Job Sheet onto the Jobs Pending Board. Note: at the Start date, Job Sheet moves to the “Current Jobs Board”, at end date it is removed for post-job processing.</td>
</tr>
<tr>
<td>34.4</td>
<td>Record Invoice number</td>
<td>JobA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Write</td>
<td>The Invoice number is recorded on the Job Booking Sheet prior to archiving</td>
</tr>
<tr>
<td>Job Status</td>
<td>Job</td>
<td>Start Date</td>
<td>End Date</td>
<td>Action</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----</td>
<td>------------</td>
<td>----------</td>
<td>--------</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>JobA</td>
<td>(CI 23) Pending Jobs Board</td>
<td>Write</td>
<td>delete Job Sheet from Pending Jobs Board at start date</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>JobA</td>
<td>(CI 24) Current Jobs Board</td>
<td>Write</td>
<td>delete Job Sheet from Current Jobs Board at end date</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Emp A</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Read</td>
<td>receive from JobA</td>
<td></td>
</tr>
<tr>
<td>17.1</td>
<td>Emp A</td>
<td>(CI 16) Quick Reference Board</td>
<td>Read</td>
<td>Preference given to employees located near job, to minimise travel &amp; LAHA costs to the client. NOTE: Implies usefulness of some kind of Travel Cost &amp; LAHA Cost &quot;ready reckoner&quot; to assist in selection of Employee (by location) to minimise Client costs. Travel bonus is calculated as a function of distance – infer use for an automated distance calculator NOTE: as for 16.1, implies the need to keep a live &quot;current location&quot; field on the Quick Reference Board</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>Emp A</td>
<td>(CI 43) GoogleMaps®</td>
<td>Access Online</td>
<td>as necessary, calculate travel time using GoogleMaps</td>
<td></td>
</tr>
<tr>
<td>??</td>
<td>Emp A</td>
<td>travel guideline ...</td>
<td>calculate</td>
<td>If under certain distance, Employee receives a Travel bonus payment based on distance. Beyond that distance, Employee is expected to stay near the site and receives a set daily LAHA allowance. Record on Selection Checklist</td>
<td></td>
</tr>
<tr>
<td>17.2</td>
<td>Give selection preference by site familiarity For multi-worker jobs, strive to place an inexperienced employee under an experienced one.</td>
<td>Emp A</td>
<td>(CI 14) Employee File</td>
<td>Read</td>
<td>For job requiring multiple employees, ensure that at least one has prior site familiarity wherever possible, and give others opportunity to become familiar. NB: Clients request close &amp; experienced workers; the Business wants employees to build experience. Employees want to increase opportunities for placement. NOTE: implies usefulness of noting which employees can act (and have acted) as 'mentors' – recorded on employee file but somehow available in Quick Reference Board. NOTE: implies the need for a LISTING of job types and conditions – used in the Experience Register, updated by job placement &amp; review, reflected in the Quick Reference Board. NOTE: implies some usefulness in tracking up-to-date experience (for specific job types &amp; conditions).</td>
</tr>
<tr>
<td>16.2</td>
<td>Ascertain qualification requirements</td>
<td>Emp A</td>
<td>(CI 20) Job Scope Report</td>
<td>Read</td>
<td>Identify nature of the job – used to select Employees. Suitable and current qualifications are an ABSOLUTE pre-condition. Pertinent experience is preferred. Opportunity in (multi-person jobs) to allow employees to enhance their experience profile (under guidance from experienced seniors) is encouraged. NOTE: perhaps a series of Flags to indicate multi-person jobs. NOTE: in the experience register – there may be need to identify key TYPES of work site and work such that experience can be easily tracked – and opportunities to acquire experience taken into account.</td>
</tr>
<tr>
<td>18.1</td>
<td>Check Employee Status</td>
<td>Emp A</td>
<td>(CI 6) Quick Reference Board</td>
<td>Read</td>
<td>Green employees are clear – Amber need mentoring &amp;/or assessment – Red to be avoided at all costs. NOTE: 3-state grading is arbitrary and imprecise. A computerised system could enhance under more than one dimension (eg: Location (auto-calculated in real time from Job location: Mentoring capacity: nature of Job (from experience register &amp; Job Scope)).</td>
</tr>
<tr>
<td>20.1</td>
<td>Select candidate</td>
<td>Emp A</td>
<td>Unspecified candidate Instrument some temporary listing of candidate employee(s)</td>
<td>Cache</td>
<td>Following preceding checks, a candidate employee is selected for further checks. Possibly not a &quot;doing&quot; but a consequence. NOTE: suggests a Flag – perhaps flag Employee or append employeeID to Job Sheet – pending confirmation of selection.</td>
</tr>
</tbody>
</table>
## Appendices

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>21.1</strong> Check Qual’s Emp A (CI 18) Job Booking Sheet</td>
<td>Ascertain Qualifications required for the job. NOTE: suggests usefulness of a listing of qualifications vs job types. NOTE: some clients specify the qualifications needed, still need to be confirmed by the Business.</td>
</tr>
<tr>
<td><strong>21.2</strong> Confirm Qual’s Emp A (CI 14) Employee File</td>
<td>Confirm candidate Employee holds appropriate current Qualifications.</td>
</tr>
<tr>
<td><strong>21.3</strong> Pass Qual’s Check Emp A (CI 14) Suitability checklist</td>
<td>Read</td>
</tr>
<tr>
<td><strong>21.4</strong> Fail Qual’s Check Emp A temporary listing of candidate employee(s)</td>
<td>Cache</td>
</tr>
<tr>
<td><strong>19.1</strong> Check Current Job Board Emp A (CI 24) Current Job Board [a folder of Job Sheets]</td>
<td>Read</td>
</tr>
<tr>
<td><strong>19.2</strong> Check Pending Job Board Emp A (CI 23) Pending Job Board [a folder of Job Sheets]</td>
<td>Read</td>
</tr>
<tr>
<td>?? Pass check Emp A (CI 14) Suitability checklist</td>
<td>Write</td>
</tr>
<tr>
<td>?? Fail Check Emp A temporary listing of candidate employee(s)</td>
<td>Cache</td>
</tr>
<tr>
<td><strong>22.1</strong> Check Experience Emp A (CI 20) Job Scope Report</td>
<td>Read</td>
</tr>
<tr>
<td><strong>22.2</strong> Confirm Experience Emp A (CI 33) Employee Experience Register</td>
<td>Read</td>
</tr>
<tr>
<td><strong>22.3</strong> Pass Experience Check Emp A (CI 14) Suitability checklist</td>
<td>Write</td>
</tr>
</tbody>
</table>
### Appendices

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>22.4</strong></td>
<td><strong>Fail Experience Check</strong>&lt;br&gt;Emp A temporary listing of candidate employee(s) Cache&lt;br&gt;<strong>If a multi-employee job, might be permitted if mentored by more experienced employee. <strong>NOTE:</strong> if selected then should update experience register (if job completed satisfactorily) – may impact on Quick Reference ‘colour’ status</strong> ELSE if check failed, delete temporary listing of candidate employee(s). Return to SIN 20.1 &amp; select new candidate. If none available, client’s job may have to be refused.**</td>
</tr>
</tbody>
</table>

| **23.1** | **Confirm Medical Cat-1 status**<br>Emp A (CI 14) Employee File Read<br>A simple gateway, irrespective of Job Scope. Employee must hold current medical clearance as issued by a government authorised medical testing agency |

| **??** | **Pass check**<br>Emp A (CI 14) Suitability checklist Write<br>**Record Pass** |

| **23.2** | **Fail Medical Cat-1 status: Reject Candidate**<br>Emp A temporary listing of candidate employee(s) Cache<br>**Cat-1 not current, cannot use that employee. Return to SIN 20.1 & select new candidate. If none available, client’s job may have to be refused.** |

| **23.2** | **Fail Medical Cat-1 status: Revise Status**<br>Emp A (CI 6) Quick Reference Board Write<br>**employee goes to immediate RED status** |

| **23.3** | **Fail Medical Cat-1 status: Flag for review**<br>Emp A (CI 14) Employee File Communiqué<br>**Employee file brought to attention of CSA, CFO & ELO** |

| **24.1** | **Confirm Induction**<br>Emp A (CI 14) Employee File Read<br>A simple gateway, irrespective of Job Scope. Employee must have passed through basic Employee Induction procedure. |

| **??** | **Pass check**<br>Emp A (CI 14) Suitability checklist Write<br>**Record Pass** |

| **??** | **Pass Check: advise FtgA**<br>Emp A (CI 14) Suitability checklist Communiqué |

| **24.3** | **Fail Induction: Reject Candidate**<br>Emp A temporary listing of candidate employee(s) Cache<br>**If Induction not recorded, employee cannot be used. Return to SIN 20.1 & select new candidate. If none available, client’s job may have to be refused.** |

| **24.2** | **Fail Induction: Rapid Induct**<br>CSA (CI 42) Employee Induction Package Read<br>Physical presence at Business premises required. **If time permits, Employee may be Inducted prior to commencement of duties. Requires personal & manual Induction procedure.** |

| **??** | **call employee in**<br>CSA (CI 14) Employee File Communiqué<br>**direct Employee to attend Main Office** |

| **??** | **Pass rapid induct**<br>CSA (CI 14) Employee File Write<br>record Induction status |

| **??** | **advise EmpA to proceed**<br>CSA Communiqué<br>**advise EmpA to proceed** |

<p>| <strong>24</strong> | <strong>receive advise from EmpA</strong>&lt;br&gt;FtgA (CI 14) Suitability checklist Read&lt;br&gt;<strong>obtain data to perform check</strong> |
| 25.1 | Estimate job duration | FtgA | (CI 20) Job Booking Sheet | Read | NB, only the FtgA has had the specific training to use FAID® and holds logon access | Obtain the work hours and days for the job, inclusive of estimated Travel Time. NB: Travel Time is a function of distance. Travel time is a factor in Fatigue calculations. NOTE: A Travel Time calculator, based on the job location and the employees expected start location (usually home) would be useful. |
| 25.2 | Check FAID® | FtgA | (CI 25) FAID® 3rd party software app. | Calculate | 3rd party FAID® estimates the level of fatigue. Employee may not work under risky levels of fatigue – may need to be relieved or refused placement if at risk. NOTE: system could format data to enhance using FAID®. Future versions of FAID® may allow direct data exchange, effectively linking into the system as a whole. |
| 25.3 | Advise JobA of FAID® status | FtgA | Telephone, email, personal conversation etc. | Communicate | Advise the JobA of the result of the FAID® check |
| 25.4 | Fail FAID® Check | FtgA | temporary listing of candidate employee(s) | Cache | Extra rest periods may be affordable and/or rotation with other employees. ELSE if fatigue check failed, delete temporary listing of candidate employee(s). Return to SIN 20.1 &amp; select new candidate. If none available, client’s job may have to be refused. |
| 26.1 | Read Duration | Emp A | (CI 20) Job Booking Sheet | Read | Read off travel time and job duration. |
| 26.2 | Confirm availability | Emp A | (CI 14) Suitability checklist | Write | A simple gateway. Employee must declare themselves available. |
| 26.3 | Fail Availability. | Emp A | temporary listing of candidate employee(s) | Cache | If check failed Return to SIN 20.1 &amp; select new candidate. If no replacement available, and/or client refuses to job-share, the client’s job may have to be refused. |
| 27.1 | Read pay rate, travel &amp; LAHA | Emp A | (CI 18) Job Booking Sheet | Read | EmpA reads relevant details from Job Booking Sheet (CI 18) |
| 27.2 | Read Employee Contact details | Emp A | (CI 14) Employee File | Read | NB: when in conjunction with other employee checks – probably not necessary to re-read contact details each time |
| 27.3 | Check with employee | Emp A | (CI 18) Job Booking Sheet | Communiqué | discuss payments and bonuses |
| 27.4 | Confirm pay rates OK | Emp A | (CI 14) Suitability checklist | Write | Note employee’s consent to the proposed reimbursement scheme. |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>27.4</td>
<td>Fail pay rates</td>
<td>Emp A</td>
<td>temporary listing of candidate employee(s)</td>
<td>Cache</td>
</tr>
<tr>
<td>28.1</td>
<td>Read Employee Contact details</td>
<td>Emp A</td>
<td>(CI 14) Employee File</td>
<td>Read</td>
</tr>
<tr>
<td>28.2</td>
<td>Check D&amp;A, Fatigue, PPE</td>
<td>Emp A</td>
<td>Telephone</td>
<td>Communiqué</td>
</tr>
<tr>
<td>28.3</td>
<td>Confirm D&amp;A, Fatigue, PPE</td>
<td>Emp A</td>
<td>(CI 14) Suitability checklist</td>
<td>Write</td>
</tr>
<tr>
<td>28.4</td>
<td>Fail D&amp;A, Fatigue, PPE</td>
<td>Emp A</td>
<td>temporary listing of candidate employee(s)</td>
<td>Cache</td>
</tr>
<tr>
<td>??</td>
<td>advise of selected employee(s)</td>
<td>Emp A</td>
<td>temporary listing of candidate employee(s)</td>
<td>Communiqué</td>
</tr>
<tr>
<td>15.1</td>
<td>Expert Review of Scope details</td>
<td>CSA</td>
<td>(CI 20) Job Scope Report</td>
<td>Flag</td>
</tr>
<tr>
<td>19.3</td>
<td>Over-ride placement &amp; re-allocate (i)</td>
<td>CSA</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Write (ie: change, modify)</td>
</tr>
<tr>
<td>19.4</td>
<td>Over-ride placement &amp; re-allocate (ii)</td>
<td>CSA</td>
<td>(CI 22) Job Sheet</td>
<td>Write (changes to (CI 18) Job Booking Sheet and (CI 22) Job Sheet)</td>
</tr>
<tr>
<td>19.5</td>
<td>Over-ride placement &amp; re-allocate (iii)</td>
<td>CSA</td>
<td>(CI 24) Current Job Board</td>
<td>Write (ie: change, modify, move instance(s) of (CI 22) Job Sheet)</td>
</tr>
<tr>
<td>19.6</td>
<td>Over-ride placement &amp; re-allocate (iv)</td>
<td>CSA</td>
<td>(CI 23) Pending Job Board</td>
<td>Write (ie: change, modify, move instance(s) of (CI 22) Job Sheet)</td>
</tr>
<tr>
<td>??</td>
<td>Record employee(s) name(s) on Job Sheet</td>
<td>JobA</td>
<td>(CI 22) Job Sheet</td>
<td>Write</td>
</tr>
<tr>
<td>??</td>
<td>append suitability checklist to Job booking Sheet</td>
<td>JobA</td>
<td>(CI 14) Suitability checklist &amp; (CI 18) Job Booking Sheet</td>
<td>append</td>
</tr>
<tr>
<td>28.5</td>
<td>Generate (CI 21) Job Letter</td>
<td>JobA</td>
<td>(CI 21) Job Letter</td>
<td>Extract relevant Job details and Print Job Letter</td>
</tr>
<tr>
<td>??</td>
<td>Send Job Letter</td>
<td>JobA</td>
<td>(CI 21) Job Letter</td>
<td>Communiqué</td>
</tr>
<tr>
<td>??</td>
<td>advise Client of selected employee(s)</td>
<td>CLO</td>
<td>advise Client of Employee? (right of refusal)</td>
<td>Communiqué</td>
</tr>
<tr>
<td>30</td>
<td>Confirm job rates with Client</td>
<td>CLO</td>
<td>(CI 2) Client File</td>
<td>Communiqué</td>
</tr>
<tr>
<td>??</td>
<td>receive Job Sheet from JobA on completion</td>
<td>CLO</td>
<td>(CI 22) Job Sheet</td>
<td>Read</td>
</tr>
<tr>
<td>31.1</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 22) Job Sheet</td>
<td>Read</td>
</tr>
<tr>
<td>31.2</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 2) Client File</td>
<td>Read</td>
</tr>
<tr>
<td>31.3</td>
<td>Contact Client</td>
<td>CLO</td>
<td>(CI 34) Client review</td>
<td>Communiqué</td>
</tr>
<tr>
<td>31.3</td>
<td>Record feedback on employee</td>
<td>CLO</td>
<td>(CI 4) Employee File</td>
<td>Write</td>
</tr>
<tr>
<td>??</td>
<td>advise CSA</td>
<td>CLO</td>
<td>(CI 4) Employee File</td>
<td>Communiqué</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>29.1</td>
<td>Record Location</td>
<td>Emp A</td>
<td>(CI 18) Job Booking Sheet</td>
<td>Read</td>
</tr>
</tbody>
</table>
| 29.2 | Record Location | Emp A | (CI 33) Employee Experience Register | Write | Record the location in the Employees experience register  
NOTE: should just be some sub-section of their individual Employee File (CI 4) |
| 30.1 | Record Work Nature | Emp A | (CI 20) Job Scope Report (part of Job Booking Sheet) | Read | At the successful conclusion of a job, read off the location  
NOTE: useful to have a standard listing of experience/work types |
| 30.2 | Record Work Nature | Emp A | (CI 33) Employee Experience Register | Write | Record the location in the Employees experience register  
NOTE: this should really just be some sub-section of their individual Employee File (CI 4) |
| 31.4 | Contact Senior Employee | Emp A | (CI 22) Job Sheet | Read | Identify mentoring/ supervising senior Employee (if appropriate) |
| 31.5 | Contact Senior Employee | Emp A | (CI 4) Employee File | Read | Read senior employee contact details |
| 31.6 | Contact Senior employee | Emp A | Telephone etc | Communiqué | Elicit feedback on junior (mentored/supervised) employee(s) |
| 31.7 | Record Feeback on employee | Emp A | (CI 4) Employee File | Write | Record feedback from client on employee performance and service |
| ?? | advise CSA | Emp A | (CI 4) Employee File | Communiqué | advise CSA of Employee completion and feedback |
| ?? | receive advise from EmpA | CSA | (CI 4) Employee File | Read | receive notice |
| 31.8 | Review Status | CSA | (CI 4) Employee File | Read | CSA may decide to alter Employee “colour” status |
| ?? | advise EmpA | CSA | email etc | Communiqué | instruct EmpA to make change as necessary |
| 31.9 | Update Status | Emp A | (CI 6) Quick Reference Board | Write | Adjust colour status at the direction of the CSA |
| 32.1 | Contact Client | CLO | (CI 22) Job Sheet | Read | Identify Client for a completed job |
| 32.2 | Contact Client | CLO | (CI 2) Client File | Read | Read client contact details |
| 32.3 | Contact Client | CLO | (CI 34) Client review | Communiqué | Confirm satisfactory completion of job and adjust charges for any variations in hours, duration etc.  
NOTE: re-contacting the client seems irrelevant, or possibly redundant after SIN 31.3 |
| 35.1 | Generate & Print hardcopy invoice | CFO | (CI 26) MYOB® 3rd party software | Output, (CI 29) Invoice | The CFO prints off a copy of the invoice |
| 35.2 | Dispatch Invoice | CFO | (CI 29) Invoice | Physical Dispatch | The AA dispatches the Invoice with the daily Mail  
NOTE: the client postal address is included on the invoice by the MYOB® software.  
NOTE: production of the envelope etc are trivial “offline” tasks not recorded in this analysis. |
### 36.1 Check for Payment
- **Gen A**
- **Incoming Mail Read**
- The AA checks for inbound payments from clients

### 36.2 Advise CFO
- **Gen A**
- **Communiqué**
- The AA advises and forwards all payments to the CFO

### 36.3 Record Payment
- **CFO (CI 26)**
- **MYOB® 3rd party software**
- **Write**
- Payment instance recorded

### 36.4 Process Payment
- **CFO (CI 31)**
- **Online banking**
- **Write**
- CFO may process payment via online banking (if appropriate) ELSE Manually during weekly banking activity

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>36.1</td>
<td>Check for Payment</td>
<td>Gen A</td>
<td>Incoming Mail Read</td>
<td>The AA checks for inbound payments from clients</td>
</tr>
<tr>
<td>36.2</td>
<td>Advise CFO</td>
<td>Gen A</td>
<td>Communiqué</td>
<td>The AA advises and forwards all payments to the CFO</td>
<td></td>
</tr>
<tr>
<td>36.3</td>
<td>Record payment</td>
<td>CFO</td>
<td>(CI 26) MYOB® 3rd party software</td>
<td>Write</td>
<td>Payment instance recorded</td>
</tr>
<tr>
<td>39</td>
<td>36.4</td>
<td>Process payment</td>
<td>CFO</td>
<td>(CI 31) Online banking</td>
<td>CFO may process payment via online banking (if appropriate) ELSE Manually during weekly banking activity</td>
</tr>
</tbody>
</table>

**Table G1: Activity Conflation**
Appendix H: Activity Shells

As part of the Analysis phase of the ATSA method, the Analyst conflates a set of Activities from SINs. These are assembled according to the rules described in chapter 4: that each Activity may only have the one common Subject, and that the SINs comprising the Activity must be consistent with a single Object.

Information available to the Analyst is unlikely to be AT terms. Through iterative consultation, the Analyst may refine, combine, split or even add elements to achieve viable Activities. An example would be the removal of specific Activities in which a junior staff member sends or receives external correspondence. It is adequate to record that some Role “sends” an instrument, as the junior staffer’s efforts are perfunctory and mechanical in this respect. Strictly speaking, these small external Activities should be recorded until and unless the Analyst and Stakeholders mutually agree that they may be subsumed.

Any such analysis is subjective, therefore, analysts may differ in their conclusions. Each Activity is then described as the seven Engström nodes. The pro forma shell is repeated here for convenience.

<table>
<thead>
<tr>
<th>ACTIVITY No.</th>
<th>Activity Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBJECT</td>
<td>the Role which ‘does’ this Activity</td>
</tr>
<tr>
<td>OBJECT</td>
<td>what the Activity wants to achieve (motive)</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>other Roles involved in this activity (recipients &amp; providers)</td>
</tr>
<tr>
<td>DivLAB</td>
<td>resources needed from others (often an Instrument)</td>
</tr>
<tr>
<td>RULES</td>
<td>constraints, conditions (things the Activity must satisfy)</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>actual result ~ effects other Activities (often an Instrument)</td>
</tr>
<tr>
<td>TOOL</td>
<td>psychological &amp; physical facilitators (including Instruments)</td>
</tr>
</tbody>
</table>

Table H1: Model Activity Shell
### New Client Process

#### ACTY - 01 Initiate Client Application Form

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>CLO (Client Liaison Officer) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Receive and record initial details of an external potential client. Ultimate motivation is to add a new Client, bringing new Job requests, so the Business can expand.</td>
</tr>
</tbody>
</table>
| COMMUNITY | External Community of Potential Clients  
Potential Client’s credit Referees  
CFO (Chief Financial Officer) : [Current Position: MD-Admin]  
GenA (General Administrator) : [Current Position: AA] |
| DivLAB | Notice from Potential Client received (maybe via GenA). [CLIENT]  
Contact details of Potential Client’s credit referees elicited, may require secondary contact initiated by CLO.  
Referral report from Potential Client’s credit referees required. [REF] |
| RULES | Must elicit details of Client (name, contact details etc).  
Must elicit contact details for credit Referees then must contact credit Referees and elicit answers to reference report questions.  
Must record all these, then send a new specific instance of (CI 1) Client Application Form to the CFO. |
| OUTCOME | (CI 1) Client Application Form (details and referee report) sent to CFO. [2] |
| TOOL | (CI 1) Client Application Form  
Letter, Fax, Telephone, eMail, Personal Conversation etc. |

#### ACTY - 02 Evaluate New Client Application

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>CFO (Chief Financial Officer) : [Current Position: MD-Admin]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Evaluate acceptability of new Client, establish a charge rate and establish MYOB account. Ultimate motivation is to add a new Client, bringing new Job requests, so the Business can expand.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>CLO (Client Liaison Officer) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>(CI 1) Client Application Form (with details and referee report) from CLO.</td>
</tr>
</tbody>
</table>
| RULES | (CI 1) Client Application Form from CLO must contain sufficient details. [1]  
CFO may over-rule credit reference reports, and has final say on acceptance of Potential Client. |
| OUTCOME | If accepted, instruct CLO to initiate a new (CI 2) Client File. [3]  
If accepted, select a charge Rate (CI 15) and advise CLO. [3]  
If accepted, create new (CI 26) MYOB® account entry. [MYOB] |
| TOOL | (CI 1) Client Application Form  
(CI 15) Rates Schedule  
(CI 26) MYOB® 3rd party software  
Telephone, eMail, Personal Conversation etc. |
**ACTY - 03**  Initiate New Client File  

**SUBJECT**  
CLO (Client Liaison Officer) : [Current Position: Office Manager]

**OBJECT**  
At the instruction of the CFO, initiate new instance of (CI 2) Client File. Ultimate motivation is to add a new Client, bringing new Job requests, so the Business can expand.

**COMMUNITY**  
CFO (Chief Financial Officer) : [Current Position: MD-Admin]

**DivLAB**  
Instruction to proceed (CI 2) from CFO. [2]  
Charge Rate decision (CI 15) from CFO. [2]

**RULES**  
Transform the (CI 1) Client Application Form, together with the data from the CFO, into an instance of (CI 2) Client File, to be filed and stored.

**OUTCOME**  
Specific instance of (CI 2) Client File, filed and stored. [FLDR ]

**TOOL**  
(CI 1) Client Application Form  
(CI 15) Rates Schedule  
(CI 2) Client File

---

**New Employee Process**

**ACTY - 04**  Receive Employment Application  

**SUBJECT**  
HumA (Human Resources Administrator) : [Current Position: AA]

**OBJECT**  
Receive and confirm new employee applications. Ultimate motivation is to add a new Employee, enabling service to more Jobs, so the Business can expand.

**COMMUNITY**  
External Community of Potential Employees.  
GenA (General Administrator) : [Current Position: AA]  
CSA (Chief Safety Auditor) : [Current Position: MD-Opns]

**DivLAB**  
Initial contact from Potential Employee received (~CI 3). [EMP]

**RULES**  
Applications are unsolicited and often unstructured. Some interpretation and/or elicitation may be required. Record on (CI 3) Employment Application.  
Accurate (confirmable) details required from Potential Employee  
Record of all applications retained in (CI 9) Employment Enquiries Folder.  
Confirm address before posting (CI 8) Employment Information Pack.

**OUTCOME**  
(CI 8) Employment Information Pack sent to applicant. [EMP]  
(CI 3) Employment App’n sent to (CI 9) Employment Enquiries Folder [FLD]

**TOOL**  
(CI 3) Employee application  
(CI 9) Employment Enquiries Folder  
(CI 8) Employment Information Pack  
Letter, Fax, Telephone, eMail, Personal Conversation etc.
### ACTY - 05 Employment Suitability Checks

<table>
<thead>
<tr>
<th><strong>SUBJECT</strong></th>
<th>HumA (Human Resources Administrator) : [Current Position: AA]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OBJECT</strong></td>
<td>Collect data to facilitate evaluation of potential employee’s suitability. Ultimate motivation is to add a new Employee, enabling service to more Jobs, so the Business can expand.</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>Potential Employee. Potential Employee’s Referees. CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Read, or elicit, Potential Employee’s Referee details.(3) [FLDR] Must contact two or more referees and elicit responses to standard questions from (CI 7) Employment Suitability Form. Must establish contact with potential employee for further checks.</td>
</tr>
</tbody>
</table>
| **RULES** | Must have Potential Employee Referee details.(*)
Contact potential employee to confirm tickets, medical, PPE. Tickets, Medical Status and PPE are all pre-requisites to suitability.
If checks passed, must advise CSA and forward Employment Suitability Form and Employment |
| **OUTCOME** | Commence an instance of (CI 7) Employment Suitability Form. [Emp] Record referee report results on Employment Suitability Form. Send Employment Suitability Form to CSA for approval. [6] |
| **TOOL** | (CI 3) Employment Application (CI 17) Employment Suitability Form Letter, Fax, Telephone, eMail, Personal Conversation etc. |

(*) In the absence of referee details, there is an exception process, conducted “offline” by the CSA whereby this step can be bypassed.
<table>
<thead>
<tr>
<th>ACTY - 06</th>
<th>Evaluate Potential Employee</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>Review application and referral reports (*) to evaluate potential employee. Ultimate motivation is to add a new Employee, enabling service to more Jobs, so the Business can expand.</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>Community of Industry Contacts &amp; Experienced Senior Employees. (*) Potential Employee.</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Must have contact details and Referee reports (3).</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>Tickets, medical and PPE are pre-requisites. Referee reports may be over-ruled (*). CSA may consult with potential employee, senior employees with personal knowledge and/or industry contacts in lieu of referral reports. Final suitability decision and initial status and condition decided by CSA. Default initial status is AMBER. Less experienced new employee may require supervision.</td>
</tr>
<tr>
<td><strong>OUTCOME</strong></td>
<td>Decision on suitability recorded on (CI 7) Employment Suitability Form, including Status and possible need for supervision. JobA advised and Employment Suitability Form sent. [7]</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>(CI 7) Employment Suitability Form Letter, Fax, Telephone, eMail, Personal Conversation etc. (*) In the absence of referee details, there is an exception process, conducted “offline” by the CSA whereby this step can be bypassed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTY - 07</th>
<th>Initiate Employee File</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>HumA (Human Resources Administrator) : [Current Position: AA]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>Enter new employee into system so they’re available for job selection. Ultimate motivation is to add a new Employee, enabling service to more Jobs, so the Business can expand.</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>New Employee CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>CSA must have approved new employee before any further processing. (CI 7) Employment Suitability Form details transferred to new specific instance of (CI 4) Employee File.</td>
</tr>
<tr>
<td><strong>OUTCOME</strong></td>
<td>New (CI 6) Quick Reference Board entry, including status colour. [BRD] PM advised of new employee and send (CI 4) Employee File. [8] FtgA advised of new employee and send (CI 4) Employee File. []</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>(CI 7) Employment Suitability Form (CI 4) Employee File (CI 6) Quick Reference Board</td>
</tr>
</tbody>
</table>
### ACTY - 08  Establish New Employee in Financial Systems

**SUBJECT**  
PM (Pay Master) : [Current Position: MD-Admin]

**OBJECT**  
Establish new employee financial details to allow payment for jobs done. Ultimate motivation is to add a new Employee, enabling service to more Jobs, so the Business can expand.

**COMMUNITY**  
New Employee  
HumA (Human Resources Administrator) : [Current Position: AA]

**DivLab**  
Receive (CI 4) Employee File from HumA. [6] (read contact details)  
Contact Employee to confirm financial details as necessary

**RULES**  
Contact new employee as necessary to confirm financial details and preferences (union membership, superannuation etc). Financial system MUST only allow payments to the correct banking accounts etc.

**OUTCOME**  
New entries (4) in the MYOB® and Online Banking systems.  
(CI 4) Employee File in Employee Folder for future reference

** TOOL**  
(CI 4) Employee File  
(CI 26) MYOB®  3rd party software  
(CI 30) Online Banking

### ACTY - 09  Establish New Employee

**SUBJECT**  
FtgA (Fatigue Administrator) : [Current Position: AA]

**OBJECT**  
Establish new employee fatigue monitoring entry. Ultimate motivation is to add a new Employee, enabling service to more Jobs, so the Business can expand.

**COMMUNITY**  
HumA (Human Resources Administrator) : [Current Position: AA]

**DivLab**  
Receive (CI 4) Employee File from HumA. [6]

**RULES**  
All employees must have fatigue evaluated under FAID® prior to job placement.  
This Activity is not critically urgent; however, establishment in FAID® must occur prior to selection of new employee for any jobs.

**OUTCOME**  
New entry (4) in (CI 25) FAID®.

** TOOL**  
(CI 14) Employee File  
(CI 25) FAID®  3rd party software app.
## Job Booking

<table>
<thead>
<tr>
<th><strong>ACTY - 10</strong></th>
<th>Solicit Jobs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>GenA (General Administrator) : [Current Position: AA]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>Quickly identify any opportunity for the Business to service a Job. Ultimate motivation is to book a Job for the Business to service with its skilled specialist employee(s).</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>Community of Clients.</td>
</tr>
<tr>
<td></td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td></td>
<td>MO (Marketing Officer) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td></td>
<td>CLO (Client Liaison Officer) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Client places Job request.</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>GenA to check off each Client on a list drawn from the Client Folder. Every Client is to be contacted (normally by telephone) at least once a week. GenA to report progress to MO. GenA to respond promptly to all Job requests, solicited or otherwise. Promptly record Job request onto a new instance of the Job Booking Sheet.</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>Client Folder collection of all (CI 2) Client Files</td>
</tr>
<tr>
<td></td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td></td>
<td>Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>ACTY - 11</strong></th>
<th>Send Job Scope Enquiry Form</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>CLO (Client Liaison Officer) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>To promptly respond to an upcoming Job and seek scope details, so it can be serviced to the highest standard. Ultimate motivation is to book a Job for the Business to service with its skilled specialist employee(s).</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>GenA (General Administrator) : [Current Position: AA]</td>
</tr>
<tr>
<td></td>
<td>Client</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Receive (CI 18) Job Booking Sheet from GenA [10]</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>The CLO is preferred to conduct Job-specific communications with Clients, though the task may be delegated – particularly after initial contact is established.</td>
</tr>
<tr>
<td></td>
<td>[NOTE: JobA &amp; CLO roles are both currently performed by the OM]</td>
</tr>
<tr>
<td></td>
<td>Promptly send client a Job Scope Enquiry Form (checking contact details from Client File as necessary)</td>
</tr>
<tr>
<td><strong>OUTCOME</strong></td>
<td>(CI 19) Job Scope Enquiry Form send to Client</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td></td>
<td>(CI 2) Client File</td>
</tr>
<tr>
<td></td>
<td>(CI 19) Job Scope Enquiry Form</td>
</tr>
<tr>
<td></td>
<td>Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
<tr>
<td>ACTY - 12</td>
<td><strong>Complete Job Scope Report</strong></td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------</td>
</tr>
</tbody>
</table>
| **SUBJECT** | JobA (Job Administrator) : [Current Position: Office Manager]  
Client (as necessary) |
| **OBJECT** | To promptly elicit and record all details of upcoming Job, so it can be serviced to the highest standard. Ultimate motivation is to book a Job for the Business to service with its skilled specialist employee(s). |
| **COMMUNITY** | GenA (General Administrator) : [Current Position: AA]  
CFO (Chief Financial Officer) : [Current Position: MD-Admin]  
EmpA (Employee Placement Administrator) : [Current Position: OM]  
Client |
| **DivLAB** | Receive (CI 18) Job Booking Sheet from GenA [10]  
Receive completed (CI 19) Job Scope Enquiry Form from Client.  
Receive elicited details from Client  
(possibly contact for more details and/or clarification) |
| **RULES** | Promptly record (eliciting as required) as many pertinent details as possible.  
Advise CFO to initiate financial entry.  
Advise EmpA to initiate employee selection. |
| **OUTCOME** | (CI 20) Job Scope Report: record necessary Job details.  
(CI 20) Job Scope Report attached to (CI 18) Job Booking Sheet  
(*henceforth treated as one document*)  
Updated (CI 18) Job Booking Sheet send to CFO [13] |
| **TOOL** | (CI 18) Job Booking Sheet  
(CI 19) Job Scope Enquiry Form  
(CI 2) Client File  
(CI 20) Job Scope Report  
Letter, Fax, Telephone, eMail, Personal Conversation etc. |
## ACTY - 13 Establish Job in Financial Systems

<table>
<thead>
<tr>
<th><strong>SUBJECT</strong></th>
<th>CFO (Chief Financial Officer) : [Current Position: MD-Admin]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OBJECT</strong></td>
<td>Establish financial record of new Job to facilitate billing and payment. Ultimate motivation is to book a Job for the Business to service by providing skilled specialist employee(s).</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Receive (CI 18) Job Booking Sheet from JobA [12]</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>Create preliminary charge entry in MYOB® for Job (Note: do not finalise!) Advise JobA</td>
</tr>
</tbody>
</table>

### OUTCOME

(CI 28) Invoice Number generated send to JobA [14]
(CI 26) MYOB® entry created.
(CI 18) filed for future reference [28,29,30]

### TOOL

(CI 28) Invoice Number generated
(CI 26) MYOB® entry created
(CI 18) Job Booking Sheet

## ACTY - 14 Create and post Job Sheet

<table>
<thead>
<tr>
<th><strong>SUBJECT</strong></th>
<th>JobA (Job Administrator) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OBJECT</strong></td>
<td>“Activate” the Job for tracking. Ultimate motivation is to book a Job for the Business to service by providing skilled specialist employee(s).</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>CFO (Chief Financial Officer) : [Current Position: MD-Admin]</td>
</tr>
</tbody>
</table>
| **DivLAB**  | Receive (CI 28) Invoice Number From CFO [13]
Read Charge rate from (CI 2) Client File (stored in Client Folder). |
| **RULES**   | Record Invoice Number onto Job Booking Sheet.
Record charge rates onto Job Booking Sheet.
Compile “half-page summary” Job Sheet from Job Booking Sheet. |

### OUTCOME

(CI 22) Job Sheet created with pay rate and (CI 29) Invoice number recorded, posted to Pending Jobs Board.
Advise EmpA [17] of (CI 22) Job Sheet

### TOOL

(CI 28) Invoice Number
(CI 18) Job Booking Sheet
(CI 2) Client File
(CI 22) Job Sheet
(CI 23) Pending Jobs Board
## Job Status

### ACTY - 15 Track Start Date

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>JobA (Job Administrator) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Ensure Job Status is correct so Job can be serviced correctly and promptly. Ultimate motivation is to provide timely and accurate service to Client, to retain Client’s future patronage.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>-</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Read (CI 22) Job Sheet from Pending Jobs Board Read current date from calendar</td>
</tr>
<tr>
<td>RULES</td>
<td>Job Sheet shows accurate Start Date. Current date ≥ Job end date (close as possible) requires a daily check by the JobA</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>(CI 22) Job Sheet moved from Pending to Current Jobs Board.</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 22) Job Sheet (*) Start date (CI 23) Pending Jobs Board (CI 24) Current Jobs Board</td>
</tr>
</tbody>
</table>

(*) In effect this is a simple status-flag change on the Job Sheet (itself merely a representative token for the Job booking Sheet) – HOWEVER the Job Boards serve as important user interfaces, so the status-flag change MUST be accompanied by a visible change in where the Job Sheet (token) appears. The Job Boards are just Visibly Displayed Folders, containing Job Sheets according to their date status.

### ACTY - 16 Track End Date

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>JobA (Job Administrator) : [Current Position: Office Manager]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Ensure Job Status is correct so Job can be serviced correctly and promptly. Ultimate motivation is to provide timely and accurate service to Client, to retain Client’s future patronage.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>CLO (Client Liaison Officer) : [Current Position; Office Manager]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Read (CI 22) Job Sheet from Current Jobs Board Read current date from calendar</td>
</tr>
<tr>
<td>RULES</td>
<td>Job Sheet located on Current Jobs Board shows accurate End Date Current date ≥ Job end date (close as possible) requires a daily check.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>(CI 22) Job Sheet removed from (CI 24) Current Jobs Board Forwarded to CLO for post-job processing [31]</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 22) Job Sheet (*) End date (CI 24) Current Jobs Board</td>
</tr>
</tbody>
</table>

(*) In effect this is a simple status-flag change on the Job Sheet (itself merely a representative token for the Job booking Sheet) – HOWEVER the Job Boards serve as important user interfaces, so the status-flag change MUST be accompanied by a visible change in where the Job Sheet (token) appears. The Job Boards are essentially just Visibly Displayed Folders, containing Job Sheets according to their date status.
# Employee Selection

## ACTY - 17 Selection by Employee default location

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify employee(s) closest to the Job location. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Receive (CI 22) Job Booking Sheet (inc. Job Scope Report) from JobA [14] Travel Distance from (CI 43) GoogleMaps and (CI 4) Employee File Select employee(s) from Employee Folder</td>
</tr>
<tr>
<td>RULES</td>
<td>Quick Reference Board accurately records employee’s default location. Must identify closest Employee(s) by default location. Closest is defined as smallest travel time. Consult online mapping tools as necessary. Must give preference to employee with Job site experience where possible</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[18] [NOTE: cache employee ID in an undescribed temporary instrument] Record Travel (Commute OR Travel OR LAHA) onto Selection Checklist Distance recorded onto (CI 18) Job Booking Sheet for FtgA [24]</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 18) Job Booking Sheet (CI 6) Quick Reference Board (CI 4) Employee File (CI 43) GoogleMaps® Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>

## ACTY - 18 Selection by Employee Qualification

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify employee(s) with adequate level of current qualifications. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>RULES</td>
<td>Exclude any candidate without adequate current qualifications. If necessary, repeat earlier checks with relaxed criteria. [NOTE: legal requirements such as Quals and Medical cannot be relaxed]</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[19] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR returned to earlier check OR Job lost</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 18) Job Booking Sheet (CI 6) Quick Reference Board Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>
### ACTY - 19 Selection by Employee availability wrt other Jobs

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Identify available employee(s) that have no clashing prior engagements. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>RULES</td>
<td>JobA maintains Current and Pending Job Boards Exclude any candidate with an inconsistent prior engagement. [Note: travel time between engagements may be a factor] If necessary, repeat earlier checks with relaxed criteria. [NOTE: legal requirements such as Quals and Medical cannot be relaxed]</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[20] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR returned to earlier check OR Job lost</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 24) Current Job Board (CI 23) Pending Job Board Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>

### ACTY - 20 Selection by Employee Experience

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>Give preference to available employee(s) that have prior experience of the Job site. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>JobA (Job Administrator) : [Current Position: Office Manager]</td>
</tr>
<tr>
<td>RULES</td>
<td>(CI 33) Employee Experience Register is current and accurate. Give preference to employees with site experience. Where possible, in multi-employee sites, strive to allow less experienced employees to gain site experience in a junior role.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[21] [NOTE: cache employee ID in an undescribed temporary instrument] Suitability Checklist point passed OR returned to earlier check OR Job lost</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist (CI 33) Employee Experience Register (CI 4) Employee File Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
</tbody>
</table>
### ACTY - 21  Selection by Employee Medical Status

- **SUBJECT**
  
  **EmpA (Employee Placement Administrator) : [Current Position: OM]**

- **OBJECT**
  
  Identify available employee(s) that have current Medical clearance. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

- **COMMUNITY**

  **HumA (Human Resources Administrator) : [Current Position: AA]**

- **DivLAB**

  Read medical status from (CI 4) Employee File

- **RULES**

  Exclude any candidate without current medical clearance. If necessary, repeat earlier checks with relaxed criteria. **[NOTE: legal requirements such as Quals and Medical cannot be relaxed]**

  Fail results in immediate change to RED status

- **OUTCOME**

  Candidate Employee(s) identified. [may be more than one] (CI 14)-[22]

  **[NOTE: cache employee ID in an undescribed temporary instrument]**

  Suitability Checklist point passed OR returned to earlier check OR Job lost

  If fail: candidate status changed to RED on (CI 6) Quick Reference Board

- **TOOL**

  (CI 14) Suitability checklist

  (CI 4) Employee File

  (CI 6) Quick Reference Board

  Unspecified candidate Instrument, listing of candidate employee(s)

---

### ACTY - 22  Selection by Employee Induction Status

- **SUBJECT**

  **EmpA (Employee Placement Administrator) : [Current Position: OM]**

- **OBJECT**

  Identify available employee(s) that have been inducted. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

- **COMMUNITY**

  **HumA (Human Resources Administrator) : [Current Position: AA]**

- **DivLAB**

  Read induction status from (CI 4) Employee File

- **RULES**

  Exclude any candidate that has not undergone induction. Advise (CI 14) FtgA to proceed [24] and CSA [23]

- **OUTCOME**

  Candidate Employee(s) identified. [may be more than one] (CI 14)-[23]

  **[NOTE: cache employee ID in an undescribed temporary instrument]**

  Suitability Checklist point passed OR returned to earlier check OR Job lost

  OR (at CSA’s discretion) candidate may be called in for rapid induction (time permitting)

  Advise FtgA to conduct FAID check (send (CI 14) Suitability checklist) [24]

- **TOOL**

  (CI 14) Suitability checklist

  (CI 4) Employee File

  Unspecified candidate Instrument, listing of candidate employee(s)

  Telephone, eMail, Personal Conversation etc.
### ACTY - 23  Rapid Induction (exception case)

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>To allow candidate employee to become rapidly inducted, in order to be available for a Job. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to Client, Employee and Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>Candidate Employee EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Employee is acceptable in all preceeding checks</td>
</tr>
<tr>
<td></td>
<td>Employee is able to attend main office for rapid induction</td>
</tr>
<tr>
<td>RULES</td>
<td>CSA has discretionary right to permit employee to attend main office for rapid induction (time permitting).</td>
</tr>
<tr>
<td></td>
<td>Sufficient time (inc travel) must be available.</td>
</tr>
<tr>
<td></td>
<td>Employee must attend and satisfy induction criteria.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Employee passes induction (CI 42). (CI 4) Employee File updated.</td>
</tr>
<tr>
<td></td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[24]</td>
</tr>
<tr>
<td></td>
<td>[NOTE: cache employee ID in an undescribed temporary instrument]</td>
</tr>
<tr>
<td></td>
<td>Suitability Checklist point passed OR returned to earlier check OR Job lost</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 4) Employee File</td>
</tr>
<tr>
<td></td>
<td>(CI 14) Suitability checklist</td>
</tr>
<tr>
<td></td>
<td>(CI 42) Employee Induction Package</td>
</tr>
<tr>
<td></td>
<td>Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
<tr>
<td></td>
<td>Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>

### ACTY - 24  Calculate Employee Fatigue Status

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>FtgA (Fatigue Administrator) : [Current Position: AA]</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBJECT</td>
<td>To estimate the level of fatigue the employee may experience in taking the Job. No employee may be put into danger. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
<tr>
<td>COMMUNITY</td>
<td>EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td>DivLAB</td>
<td>Receive complete (CI 14) Suitability checklist including travel status</td>
</tr>
<tr>
<td></td>
<td>Receive complete (CI 18) Job Booking Sheet including duration details</td>
</tr>
<tr>
<td>RULES</td>
<td>Only the FtgA is trained and authorised to use the FAID® system</td>
</tr>
<tr>
<td></td>
<td>Employees may NOT be used if they're exposed to dangerous levels of fatigue as assessed by FAID®</td>
</tr>
<tr>
<td></td>
<td>If exposed to fatigue, extra rest periods (typically extra days of LAHA) and/or rotation with other employees may be considered.</td>
</tr>
<tr>
<td>OUTCOME</td>
<td>Candidate Employee(s) identified. [may be more than one] (CI 14)-[25]</td>
</tr>
<tr>
<td></td>
<td>[NOTE: cache employee ID in an undescribed temporary instrument]</td>
</tr>
<tr>
<td></td>
<td>Suitability Checklist point passed OR some extra rest time solution determined OR returned to earlier check OR Job lost.</td>
</tr>
<tr>
<td>TOOL</td>
<td>(CI 14) Suitability checklist</td>
</tr>
<tr>
<td></td>
<td>(CI 18) Job Booking Sheet</td>
</tr>
<tr>
<td></td>
<td>Unspecified candidate Instrument, listing of candidate employee(s)</td>
</tr>
<tr>
<td></td>
<td>Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>
### ACTY - 25  Selection by Employee Personal Availability

**SUBJECT**  
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**  
Confirm the employee is available according to their personal schedule. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

**COMMUNITY**  
JobA (Job Administrator) : [Current Position: Office Manager]  
Candidate Employee

**DivLAB**  
Read (CI 18) Job Booking Sheet  
Read (CI 4) Employee File contact details

**RULES**  
Employee must be available and willing to undertake the work.

**OUTCOME**  
Candidate Employee(s) identified. [may be more than one] (CI 14)-[26]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra rest time solution determined OR returned to earlier check OR Job lost.

**TOOL**  
(CI 18) Job Booking Sheet  
(CI 4) Employee File  
Unspecified candidate Instrument, listing of candidate employee(s)  
Letter, Fax, Telephone, eMail, Personal Conversation etc.

### ACTY - 26  Selection by Employee Agreement to Conditions and Pay

**SUBJECT**  
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**  
Confirm the employee agrees to the payments and bonuses entailed in the Job. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

**COMMUNITY**  
Candidate Employee.

**DivLAB**  
Read (CI 18) Job Booking Sheet  
Read (CI 4) Employee File contact details

**RULES**  
Pay rates conform to CFO rates decisions for Client + standard bonuses. Employee must be agreeable to the payments and bonuses.

**OUTCOME**  
Candidate Employee(s) identified. [may be more than one] (CI 14)-[27]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra rest time solution determined OR returned to earlier check OR Job lost.

**TOOL**  
(CI 18) Job Booking Sheet  
(CI 4) Employee File  
Unspecified candidate Instrument, listing of candidate employee(s)  
Letter, Fax, Telephone, eMail, Personal Conversation etc.
### ACTY - 27  
**Selection by Employee Compliance with Policies**

<table>
<thead>
<tr>
<th><strong>SUBJECT</strong></th>
<th>EmpA (Employee Placement Administrator) : [Current Position: OM]</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OBJECT</strong></td>
<td>Confirm the employee is familiar with, and agrees to comply with, Business Policies on Drugs &amp; Alcohol, fatigue and PPE. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
</tbody>
</table>
| **COMMUNITY** | Candidate Employee.  
CSA (Chief Safety Auditor) : [Current Position: MD-Opns]  
JobA (Job Administrator) : [Current Position: Office Manager] [29] |
| **DivLAB** | Read (CI 4) Employee File contact details |
| **RULES** | Employee must conform to policies.  
Employee must have opportunity to declare concerns, request clarifications or declare non-conformance. |
| **OUTCOME** | Candidate Employee(s) identified. [may be more than one] (CI 14)-[28,29]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra time for solutions (if available) OR returned to earlier check OR Job lost.  
Record employee compliance.  
Advise CSA and JobA of employee selection. |
| **TOOL** | (CI 14) Employee File  
(CI 14) Suitability checklist  
Letter, Fax, Telephone, eMail, Personal Conversation etc.  
Unspecified candidate Instrument, listing of candidate employee(s) |
<table>
<thead>
<tr>
<th>ACTY - 28</th>
<th>Review of Selection by Experience (rare exception case)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>The CSA makes final overview of the selection and has the power to override selections (without breaking legal constraints). Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.</td>
</tr>
</tbody>
</table>
| **COMMUNITY** | Client  
Candidate Employee.  
EmpA (Employee Placement Administrator) : [Current Position: OM]  
JobA (Job Administrator) : [Current Position: Office Manager] [29]  
CLO (Client Liaison Officer) : [Current Position; Office Manager] [30] |
| **DivLAB** | CSA advised of selected employee (14) and of Job Booking details (18)  
May seek to consult employee  
May seek acceptance from client |
| **RULES** | CSA may NOT override legal selection constraints (quals, medical etc). |
| **OUTCOME** | Candidate Employee(s) review. [NB: maybe more than one] (CI 14)-[29,30]  
[NOTE: cache employee ID in an undescribed temporary instrument]  
Suitability Checklist point passed OR some extra time for solutions (if available) OR returned to earlier check OR Job lost. |
| **TOOL** | (CI 18) Job Booking Sheet  
(CI 24) Current Job Board  
(CI 23) Pending Job Board  
Unspecified candidate Instrument, listing of candidate employee(s) |
| (*) | Includes the rare exception case that specific Client has requested NOT to use specific employee in their Jobs |
## ACTY - 29 Finalise Selection Documents

### SUBJECT
JobA (Job Administrator) : [Current Position: Office Manager]

### OBJECT
Combine documents to finalise employee selection. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

### COMMUNITY
CSA (Chief Safety Auditor) : [Current Position: MD-Opns]

### DivLAB
EmpA has forwarded (CI 14) Suitability Checklist. (CI 18) Job Booking Sheet is complete and accurate.

### RULES
Documentation must be resolved for all future processes. Job Letter to Employee constitutes a contract to perform the agreed duties for the agreed remuneration.

### OUTCOME
(CI 14) Suitability Checklist joined to (CI 18) Job Booking Sheet, to CLO [30] (CI 22) Job Sheet now has employee ID. (CI 21) Job Letter generated and dispatched to employee(s).

### TOOL
(CI 18) Job Booking Sheet  
(CI 22) Job Sheet  
(CI 21) Job Letter  
Unspecified candidate Instrument, listing of candidate employee(s)

## ACTY - 30 Client advised of Selection

### SUBJECT
CLO (Client Liaison Officer) : [Current Position; Office Manager]

### OBJECT
Client must be advised of the employee and conditions selected to service their Job. Ultimate motivation is to choose the right employee for each Job to provide maximum benefit to the Client, the Employee and the Business.

### COMMUNITY
Client  
DivLAB (CI 18) Job Booking Sheet is complete and accurate.

### RULES
Employee selection is finished  
Client must be advised.

### OUTCOME
Client must be advised (CI 21(?)) of the service to be provided.

### TOOL
(CI 18) Job Booking Sheet  
(CI 2) Client File  
Letter, Fax, Telephone, eMail, Personal Conversation etc.  
Unspecified candidate Instrument, listing of candidate employee(s)
Review Employee

<table>
<thead>
<tr>
<th>ACTY - 31</th>
<th>Elicit Client Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>CLO (Client Liaison Officer) : [Current Position; Office Manager]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>Elicit Clients satisfaction with Employee performance to assist in monitoring and enhancing their abilities. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>Client CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Received completed (CI 22) Job Sheet from JobA [16]</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>JobA’s daily review of dates and Current Jobs Board allow prompt notification that a Job is complete &amp; employee has completed all tasks. Client feedback may be received (or requested) at any point in the Job, but specific post-Job feedback is required to assess Employee performance.</td>
</tr>
<tr>
<td><strong>OUTCOME</strong></td>
<td>Client comments are elicited following the (CI 34) Client Review worksheet. Feedback on the Employee is recorded in the Employee File. CSA advised of Employee performance (as per Clients).</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>(CI 22) Job Sheet (CI 2) Client File (CI 34) Client review (CI 4) Employee File Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
<tr>
<td>(*)</td>
<td>Note: some smaller post-Job processes [ recording Job details, receipt of work reports from the senior Employee, copies of Briefing Reports, Risk Assessments, Safety Plans &amp; time-sheets (authenticated by the Client) for Employee payments.] These, together with final billing, payments and Customer satisfaction surveys are not covered in this Method assessment.</td>
</tr>
<tr>
<td>ACTY - 32</td>
<td>Update Employee Experience Register: Location and Work Nature</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>SUBJECT</strong></td>
<td>EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>Record details of Employees duties to assist in monitoring and enhancing their experience base. This is necessary so that future employee selections based on experience are accurate. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>-</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>(CI 22) Job Sheet notification (CI 18) Job Booking Sheet (inc. (CI 20) Job Scope Report)</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>Job is complete. The Experience register should be expressed in terms compatible with the Job Scope Request Form (to facilitate matching employees to Jobs)</td>
</tr>
<tr>
<td><strong>OUTCOME</strong></td>
<td>(CI 33) Employee Experience Register records details of completed Job's location and nature of work (following consistent taxonomy as Job Scope request form terminology)</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>(CI 18) Job Booking Sheet (inc. (CI 20) Job Scope Report) (CI 33) Employee Experience Register (part of (CI 4) Employee File)</td>
</tr>
<tr>
<td>(*)</td>
<td>Note: At the time of the analysis, the Experience Registration Process was a new and still evolving idea. Strongest indication at the time was that the register would be treated as a specific sub-section of the Employee File.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTY - 33</th>
<th>Elicit Supervisor Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SUBJECT</strong></td>
<td>EmpA (Employee Placement Administrator) : [Current Position: OM]</td>
</tr>
<tr>
<td><strong>OBJECT</strong></td>
<td>Elicit senior (supervising) senior Employee satisfaction with Employee performance to assist in monitoring and enhancing their abilities. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.</td>
</tr>
<tr>
<td><strong>COMMUNITY</strong></td>
<td>Senior (supervising) Employee. CSA (Chief Safety Auditor) : [Current Position: MD-Opns]</td>
</tr>
<tr>
<td><strong>DivLAB</strong></td>
<td>Employee was places under supervision of a more senior Employee.</td>
</tr>
<tr>
<td><strong>RULES</strong></td>
<td>Supervisors feedback assists CSA in reviewing Employee status.</td>
</tr>
<tr>
<td><strong>OUTCOME</strong></td>
<td>Employee File record supervisor feedback. CSA advised of Supervisors’s feedback (CI 33). [34]</td>
</tr>
<tr>
<td><strong>TOOL</strong></td>
<td>(CI 22) Job Sheet (CI 4) Employee File inc. (CI 33) Employee Experience Register Letter, Fax, Telephone, eMail, Personal Conversation etc.</td>
</tr>
</tbody>
</table>
### ACTY - 34  
**Review Employee ‘colour’ Status**

**SUBJECT**  
CSA (Chief Safety Auditor) : [Current Position: MD-Opns]

**OBJECT**  
The Business hopes Employees will improve and lift their colour Status. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.

**COMMUNITY**  
EmpA (Employee Placement Administrator) : [Current Position: OM]

**DivLAB**  
Advice from EmpA that review comments on Employee performance are available after the end of a Job. (CI 33)

**RULES**  
Colour Status greatly impacts on employee’s suitability for future work.

**OUTCOME**  
Determine if changes are necessary to Employee status. Advise EmpA of any change to Employee status. (CI 4) [35]

**TOOL**  
(CI 4) Employee File, inc. (CI 33) Employee Experience Register  
Telephone, eMail, Personal Conversation etc.

(*)  
CSA may choose to consult with Client, with Employee etc. These are discretionary doings that are outside the scope of any computer facilitated system.

---

### ACTY - 35  
**Update Employee colour Status as required.**

**SUBJECT**  
EmpA (Employee Placement Administrator) : [Current Position: OM]

**OBJECT**  
Record revised employee colour status in Quick Reference Board. This ensures future selections are accurately decided. Ultimate motivation is to improve the quality of the Businesses service and ensure Client satisfaction.

**COMMUNITY**  
CSA (Chief Safety Auditor) : [Current Position: MD-Opns]

**DivLAB**  
Receive advise (CI 4) from CSA [34]

**RULES**  
Colour status is a vital indicator for employee selection and the Quick reference board must be kept accurate and current.

**OUTCOME**  
Quick reference board (CI 6) updated with revised employee colour status.

**TOOL**  
(CI 4) Employee File, inc. (CI 33) Employee Experience Register  
(CI 6) Quick Reference Board  
Telephone, eMail, Personal Conversation etc.

---

**Tables H2: Conflated Activity Shells**
Appendices

Appendix I: Combined Activity Table (CAT)

<p>| Figure II: Combined Activity Table (CAT) |</p>
<table>
<thead>
<tr>
<th>ACT#</th>
<th>Role</th>
<th>desc</th>
<th>INST IN</th>
<th>from</th>
<th>INST OUT</th>
<th>to</th>
<th>Transform</th>
<th>JOINT?</th>
<th>PIPE?</th>
<th>collapsible?</th>
<th>collapsible?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CLO</td>
<td>Initiate Client Application Form</td>
<td>applicants EMP</td>
<td>receive</td>
<td></td>
<td></td>
<td>0.01, 2</td>
<td>2 create</td>
<td></td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CFO</td>
<td>Evaluate New Client Application</td>
<td></td>
<td>newMYOB MYOB</td>
<td></td>
<td></td>
<td>0.02, 3</td>
<td>3 create</td>
<td></td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CLO</td>
<td>Initiate New Client File</td>
<td>CI 02</td>
<td>2</td>
<td>0.02 Folder</td>
<td>store</td>
<td>NO</td>
<td></td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>HumA</td>
<td>Receive Employment Application</td>
<td>CI 05</td>
<td>EMP</td>
<td></td>
<td></td>
<td>0.08 Folder</td>
<td></td>
<td></td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>HumA</td>
<td>Employment Suitability Checks</td>
<td>CI 03</td>
<td>Folders</td>
<td></td>
<td></td>
<td>0.07 EMP</td>
<td></td>
<td></td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>CSA</td>
<td>Evaluate Potential Employee</td>
<td>CI 03</td>
<td>EMP</td>
<td></td>
<td></td>
<td>0.07, 7</td>
<td>7 decide</td>
<td>NO</td>
<td></td>
<td>NO</td>
</tr>
<tr>
<td>7</td>
<td>HumA</td>
<td>Initiate Employee File</td>
<td>CI 07</td>
<td>5</td>
<td></td>
<td></td>
<td>0.06, Boards</td>
<td></td>
<td></td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>PM</td>
<td>Establish New Employee in Financial Systems</td>
<td>CI 04</td>
<td>7</td>
<td>readcontacts</td>
<td>0.04 MYOB establish</td>
<td>NO</td>
<td></td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>TIGA</td>
<td>Establish New Employee</td>
<td>CI 04</td>
<td>7</td>
<td>0.04 FAID establish</td>
<td></td>
<td>NO</td>
<td></td>
<td>NO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>GenA</td>
<td>Solicit Jobs</td>
<td>CI 18</td>
<td>11</td>
<td></td>
<td></td>
<td>0.18, 12</td>
<td>12 NO</td>
<td></td>
<td>NO</td>
<td></td>
</tr>
<tr>
<td>ACTN</td>
<td>Role</td>
<td>desc</td>
<td>INST</td>
<td>from</td>
<td>to</td>
<td>Transform</td>
<td>JOINT</td>
<td>PIPE</td>
<td>collapsible?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>----</td>
<td>----------</td>
<td>-------</td>
<td>------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>CLO</td>
<td>Send Job Scope Enquiry Form</td>
<td>C1 10</td>
<td>10</td>
<td>C1 19</td>
<td>client</td>
<td>N O</td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>JobA</td>
<td>Complete Job Scope Report</td>
<td>C1 10</td>
<td>10</td>
<td>C1 10 13</td>
<td></td>
<td>N O</td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>CFO</td>
<td>Establish Job in Financial Systems</td>
<td>C1 10</td>
<td>12</td>
<td>C1 18 Folders</td>
<td></td>
<td>YES Possible</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>JobA</td>
<td>Create and post Job Sheet</td>
<td>C1 02 Folders</td>
<td>13</td>
<td>C1 22 Boards create C1 20 as keyfield</td>
<td>YES</td>
<td>N O: temporal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>JobA</td>
<td>Track Start Date</td>
<td>C1 22 Boards</td>
<td></td>
<td>C1 22 Boards</td>
<td>pending = current</td>
<td>YES</td>
<td>N O: status diff</td>
<td>N O: temporal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>JobA</td>
<td>Track End Date</td>
<td>C1 22 Boards</td>
<td></td>
<td>C1 22 Folders = active</td>
<td>current finished</td>
<td>YES</td>
<td>N O: status diff</td>
<td>N O: temporal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>EmpA</td>
<td>Selection by Employee default location</td>
<td>C1 04 Folders employIDs</td>
<td></td>
<td>C1 18 Folders</td>
<td></td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C1 18 Folders</td>
<td></td>
<td></td>
<td></td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C1 06 Boards</td>
<td>default locations</td>
<td></td>
<td></td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C1 22 14</td>
<td></td>
<td></td>
<td></td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>distance</td>
<td>Google maps</td>
<td>calculation</td>
<td></td>
<td>N O</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C1 10 Folders</td>
<td>record</td>
<td>recordDistance</td>
<td>YES</td>
<td>N O: temporal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C1 14 cache</td>
<td>record</td>
<td>decide travel status</td>
<td>YES</td>
<td>Possible</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACT #</td>
<td>Role Role</td>
<td>Role Role</td>
<td>INST IN</td>
<td>from</td>
<td>command</td>
<td>INST OUT</td>
<td>to</td>
<td>command</td>
<td>Transform</td>
<td>JOINT ?</td>
<td>PIPE ?</td>
</tr>
<tr>
<td>-------</td>
<td>-----------</td>
<td>-----------</td>
<td>--------</td>
<td>------</td>
<td>---------</td>
<td>----------</td>
<td>----</td>
<td>---------</td>
<td>-----------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>18 EmpA</td>
<td>Selection by Employee Qualification</td>
<td>CI 06 Boards</td>
<td>readBoard</td>
<td>CI 10 Folders</td>
<td>read</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 16 c Ach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>19 EmpA</td>
<td>Selection by Employee Availability with other Jobs</td>
<td>CI 23 Boards</td>
<td>readBoard</td>
<td>CI 24 Folders</td>
<td>read</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>20 EmpA</td>
<td>Selection by Employee Experience</td>
<td>CI 04 Folders</td>
<td>readFolder</td>
<td>CI 04 Folders</td>
<td>read</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>21 EmpA</td>
<td>Selection by Employee Medical Status</td>
<td>CI 04 Folders</td>
<td>readFolder</td>
<td>CI 04 Folders</td>
<td>read</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>22 EmpA</td>
<td>Selection by Employee Induction Status</td>
<td>CI 04 Folders</td>
<td>readFolder</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>23 CSA</td>
<td>Rapid Induction (exception case)</td>
<td>CI 13 Folders</td>
<td>readFolder</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>24 FigA</td>
<td>Calculate Employee Fatigue Status</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
<tr>
<td>25 EmpA</td>
<td>Selection by Employee Personal Availability</td>
<td>CI 04 Folders</td>
<td>readContacts</td>
<td>CI 04 Folders</td>
<td>read</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
<td>CI 14 Cach</td>
<td>record</td>
</tr>
</tbody>
</table>

*Indicates a temporal role.
<table>
<thead>
<tr>
<th>ACTY#</th>
<th>Role Actor</th>
<th>desc</th>
<th>INST IN</th>
<th>INST OUT</th>
<th>Transform</th>
<th>JOINT ?</th>
<th>PIPE ?</th>
<th>POSSIBLE Links or From state value?</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>Emp</td>
<td></td>
<td>C04</td>
<td>Follics</td>
<td>Enquiry</td>
<td>25</td>
<td></td>
<td>C04, rado, rado</td>
</tr>
<tr>
<td>27</td>
<td>Emp</td>
<td></td>
<td>C04</td>
<td>Follics</td>
<td>Enquiry</td>
<td>26</td>
<td></td>
<td>C04, rado, rado</td>
</tr>
<tr>
<td>28</td>
<td>CSA</td>
<td></td>
<td>C04</td>
<td>Follics</td>
<td>Client</td>
<td>27</td>
<td></td>
<td>C04, rado, rado</td>
</tr>
<tr>
<td>29</td>
<td>CLO</td>
<td></td>
<td>C04</td>
<td>Follics</td>
<td>Client</td>
<td>28</td>
<td></td>
<td>C04, rado, rado</td>
</tr>
</tbody>
</table>

Legend:
- Emp: Employee
- Follics: Follics
- C04: C04
- C14: C14
- C15: C15
- C02: C02
- C16: C16
- C18: C18
- C10: C10
- C14: C14
- C15: C15
- C16: C16
- C18: C18

Possible links or from state value:
- C04, rado, rado
- C04, rado, rado
- C04, rado, rado
- C04, rado, rado
### Figures J1: Node Reduction Heuristic (CAT variant)

<table>
<thead>
<tr>
<th>Role</th>
<th>desc</th>
<th>INST</th>
<th>from</th>
<th>action</th>
<th>INST</th>
<th>to</th>
<th>action</th>
<th>Transform</th>
<th>collapse?</th>
<th>PIPE?</th>
<th>link-to-from same Role?</th>
<th>collapse?</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>CLO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>EmpA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>EmpA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>CSA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>EmpA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Appendices*