Strategic alignment of business processes

Evan Morrison

University of Wollongong

Follow this and additional works at: https://ro.uow.edu.au/theses
STRATEGIC ALIGNMENT OF BUSINESS PROCESSES

A Dissertation Submitted in Fulfilment of
the Requirements for the Award of the Degree of

Doctor of Philosophy

from

UNIVERSITY OF WOLLONGONG

by

Evan Morrison
B. Computer Science

School of Computer Science and Software Engineering
Faculty of Engineering

2014
CERTIFICATION

I, Evan Morrison, 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 Engineering, 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.

(Signature Required)
Evan Morrison
10 January 2014
# Contents

<table>
<thead>
<tr>
<th>List of my Publications</th>
<th>iv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>vi</td>
</tr>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Outcomes and Contributions</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Thesis Structure</td>
<td>5</td>
</tr>
<tr>
<td>2 Literature Review</td>
<td>7</td>
</tr>
<tr>
<td>2.1 Formal Preliminaries</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Goal and Strategy Modeling</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Specification and Verification</td>
<td>16</td>
</tr>
<tr>
<td>2.4 Business Process Modeling and Workflow</td>
<td>18</td>
</tr>
<tr>
<td>2.5 Compliance</td>
<td>24</td>
</tr>
<tr>
<td>2.6 Similarity Modeling and Redesign</td>
<td>28</td>
</tr>
<tr>
<td>2.7 Quality of Service</td>
<td>32</td>
</tr>
<tr>
<td>2.8 Alignment</td>
<td>34</td>
</tr>
<tr>
<td>2.9 Summary</td>
<td>39</td>
</tr>
<tr>
<td>3 Process Modeling, Annotation, and Composition</td>
<td>41</td>
</tr>
<tr>
<td>3.1 Semantic Processes</td>
<td>43</td>
</tr>
<tr>
<td>3.2 Need for Semantic Process Composition</td>
<td>46</td>
</tr>
<tr>
<td>3.3 Motivating Example</td>
<td>47</td>
</tr>
<tr>
<td>3.4 Task Composition and Choreography</td>
<td>53</td>
</tr>
<tr>
<td>3.4.1 Semantic Effects and Accumulation</td>
<td>53</td>
</tr>
<tr>
<td>3.4.2 Semantical Annotated Processes</td>
<td>54</td>
</tr>
<tr>
<td>3.4.3 Semantic Process Composition</td>
<td>56</td>
</tr>
<tr>
<td>3.5 Conclusion</td>
<td>61</td>
</tr>
<tr>
<td>4 Business Process Integration</td>
<td>62</td>
</tr>
<tr>
<td>4.1 Need for Process Integration</td>
<td>63</td>
</tr>
<tr>
<td>4.2 Approaches to Integration</td>
<td>65</td>
</tr>
<tr>
<td>4.3 Annotated Process Nets</td>
<td>73</td>
</tr>
<tr>
<td>4.4 Semantic Process Net Integration</td>
<td>74</td>
</tr>
<tr>
<td>CONTENTS</td>
<td>ii</td>
</tr>
<tr>
<td>----------</td>
<td>----</td>
</tr>
<tr>
<td>4.4.1 Case Sequence Integration Criteria</td>
<td>77</td>
</tr>
<tr>
<td>4.4.2 Case Sequence Integration Methodology</td>
<td>78</td>
</tr>
<tr>
<td>4.5 Summary and Conclusions</td>
<td>79</td>
</tr>
<tr>
<td>5 Strategic Alignment</td>
<td>81</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>82</td>
</tr>
<tr>
<td>5.2 Strategy Modeling</td>
<td>83</td>
</tr>
<tr>
<td>5.2.1 Strategy Modeling Language</td>
<td>83</td>
</tr>
<tr>
<td>5.2.2 Strategy Ontology</td>
<td>85</td>
</tr>
<tr>
<td>5.2.3 Organisational</td>
<td>86</td>
</tr>
<tr>
<td>5.2.4 Strategy</td>
<td>88</td>
</tr>
<tr>
<td>5.2.5 Operational</td>
<td>90</td>
</tr>
<tr>
<td>5.2.6 Context</td>
<td>91</td>
</tr>
<tr>
<td>5.3 Strategic Alignment of Business Processes</td>
<td>94</td>
</tr>
<tr>
<td>5.4 Implementation</td>
<td>98</td>
</tr>
<tr>
<td>5.5 Conclusion</td>
<td>99</td>
</tr>
<tr>
<td>6 Organisational Compliance Management</td>
<td>101</td>
</tr>
<tr>
<td>6.1 Introduction</td>
<td>102</td>
</tr>
<tr>
<td>6.2 Motivations</td>
<td>106</td>
</tr>
<tr>
<td>6.2.1 Running Example</td>
<td>109</td>
</tr>
<tr>
<td>6.3 Measuring Degrees of Compliance</td>
<td>111</td>
</tr>
<tr>
<td>6.3.1 Compliance Assessment Framework</td>
<td>113</td>
</tr>
<tr>
<td>6.3.2 Identifying Imprecise Compliance Requirements</td>
<td>117</td>
</tr>
<tr>
<td>6.3.3 Engineering Compliance Requirements</td>
<td>117</td>
</tr>
<tr>
<td>6.3.4 Methodology</td>
<td>119</td>
</tr>
<tr>
<td>6.4 Prioritizing repair of non-compliant processes</td>
<td>122</td>
</tr>
<tr>
<td>6.5 Summary and Conclusions</td>
<td>124</td>
</tr>
<tr>
<td>7 Toolkit Development</td>
<td>125</td>
</tr>
<tr>
<td>7.1 Graph Encoding a Process Model</td>
<td>128</td>
</tr>
<tr>
<td>7.1.1 Decision Free Task Sequence</td>
<td>134</td>
</tr>
<tr>
<td>7.2 Effects, WFFs and Accumulation</td>
<td>138</td>
</tr>
<tr>
<td>7.3 QOS Preferences</td>
<td>148</td>
</tr>
<tr>
<td>8 Conclusion</td>
<td>158</td>
</tr>
<tr>
<td>8.1 Summary of contributions</td>
<td>159</td>
</tr>
<tr>
<td>8.2 Benefits and application</td>
<td>160</td>
</tr>
<tr>
<td>8.3 Shortcomings</td>
<td>161</td>
</tr>
<tr>
<td>8.4 Future work</td>
<td>161</td>
</tr>
<tr>
<td>Appendix A Case Study</td>
<td>162</td>
</tr>
<tr>
<td>A.0.1 Market</td>
<td>164</td>
</tr>
<tr>
<td>A.1 Company Overview</td>
<td>165</td>
</tr>
<tr>
<td>A.1.1 Business Opportunities</td>
<td>165</td>
</tr>
</tbody>
</table>
List of my Publications


crowd-sourcing to Verify a Radiation Oncology Ontology: a Summer Project.
In JROI - Journal of Radiation Oncology Informatics vol6 no1, 2014.

Strategic Alignment of Business Processes

Evan Morrison

A Thesis for Doctor of Philosophy

School of Computer Science and Software Engineering
University of Wollongong

Abstract

Strategic business process alignment is the practice of correlating business processes to organisational strategy, addressing problems within organisations where processes are misaligned. A business may have processes that do not contribute towards their organisational strategies. Additionally, there may be organisational strategies that are unfulfilled as there are no processes within the business that can satisfy them. Through process composition and assessing strategic alignment, businesses can ensure that all components of the organisation work towards the core goals and the company vision. The motivation of this thesis stems from a lack of formal understanding of the subject.

To develop and define concept of alignment in the space of organisational strategy, this dissertation explores the nature of strategy, business process management & composition, integration, compliance and, goal realization. It then provides deep analysis on how these fields can be connected to create a general method for alignment.

This thesis seeks to develop a methodological framework and supporting toolkit to provide a measurable assessment of alignment between a portfolio of business processes and the strategic landscape of an organisation. This can be further used to establish re-alignment in a dynamic enterprise context. A suite of tools (TextSeer) as well as a method to assess strategic alignment has been designed and developed through this thesis. These tools enable businesses to better understand their operations. This understanding allows an organisation to manage their processes more efficiently with clear process descriptions and process to strategy mappings. The thesis aids in the modeling and management of processes and business strategies by relating each process within an organisation to related organisational strategies.

Keywords: Business Process Management, Strategy, Process Integration, Compliance Management, Requirements Engineering
Chapter 1

Introduction

“Hofstadter’s Law: It always takes longer than you expect, even when you take into account Hofstadter’s Law...

DOUGLAS R. HOFSTADTER

BUSINESS process systems are adopted in companies wishing to leverage frameworks that assist choreographing operational activities for strategic advantage. This includes the coordination of human activities with services [21][36]. The use of a standardized modeling notation such as the Business Process Modeling Notation (BPMN) is not enough to guarantee the correctness of operations. This dissertation aims to investigate process composition & choreographing, integration of processes, process compliance management and process to goal realization to assure consistency in complex organisational models.

The trend in businesses is towards more complex processes that are difficult to manage (due to lifecycle, new systems, and loss of organisational knowledge) [168]. In addition, as the external complexity of organisations grows, so too does its internal complexity. This inflates the difficulty of managing the organisation and its interactions with customers, suppliers and stakeholders. Hence, there is an need for advanced information systems and process management tools that can be utilised by a less tech-savvy audience to manage this complexity. As such there is a focus in this thesis on the management of business process systems with consideration and automated tool support for process compliance, process integration, and process/goal realization to measure the effectiveness of the businesses processes.
1.1 Motivation

Within any organisation, it stands to reason that all processes should exist to achieve some strategic objective. All activities and processes should be conducted so that there is a partial realization of at least one strategic objective for a process. The purposes of aligning business processes with organisational strategies are numerous. As an example, these can include:

1. Minimization of redundant processes.
2. Ensuring that strategies are fulfilled.
3. Mitigation of resource misuse.
4. Minimization of costly compliance breaches.
5. Improving to the ability of an organisation to manage change.
6. Increasing understanding of the complexity within an organisation.
7. Bridging the knowledge gaps between strategy makers and process workers.

Existing frameworks in the requirements engineering and service oriented architecture space (E3 [5], Instal [190], GOORE [186]) attempt to provide attention to the problem of evaluating and maintaining alignment. Though these frameworks do not provide generalised systems that are applicable to a wide variety of organisations. In addition, none of the above frameworks provides methods for managing alignment in the wake of non-conformance or non-compliance. Strategic business process alignment is the practice of correlating business processes to organisational strategies taking into consideration process choreography, integration, compliance and realization.

Real-life compliance requirements are often imprecise, i.e., compliance checking does not generate Boolean answers. Work to date in the area of compliance management has resulted in frameworks [146] that link legislative requirements to business rules, policies and processes. A compliance requirement is structurally and sometimes semantically related to a goal; degree of compliance, for this reason, leads to a degree of alignment measurement. It is thus, incredibly important to incorporate a compliance management system into any system for measurement of alignment.

Process integration takes as input a set of process designs that have approximately similar functionality (i.e., processes that describe alternative ways of doing the same
thing). The output of the integration operation is a canonical process as similar as possible to the original designs. The resultant processes must have all of the functionality of the original processes. Existing integration frameworks by Morrison et. al. [147] have been generalised further in this dissertation for inclusion in the alignment framework presented. In the context of strategic process alignment these measures allow us to achieve the following kinds of analysis. Process Rationalization: Discovery of alternative processes that realize the same strategy, and then rationalization of these processes into a single economic process (optimized to some objective function). Process substitutability: If the process that achieved a strategy needs to be retired, it can be replaced with the nearest and most similar process. Change management: If a process needs to change because of a shift in the strategic landscape, distance measures help decide on the best and most consistent process to use as a substitution. By incorporating process integration into an alignment framework it becomes possible to manage change more effectively and to minimize redundancies across the business.

To enable further reasoning over organisation strategies during alignment a strategy modeling language has been described [124]. This dissertation has extended on this language. An ontology for strategy description has been developed to address both the functional aspect and non-functional properties of strategies.

1.2 Outcomes and Contributions

This thesis shows that a methodological framework and supporting toolkit can support the assessment of alignment between a portfolio of business processes and the strategic landscape of an organisation. The framework can be further used to establish re-alignment in a dynamic enterprise context. The aim of which is to provide support for organisations struggling to manage complexity.

Although business processes have been around for many years, there does not exist a widely accepted method for assessing alignment between processes and organisational strategies. To prove this thesis, a suite of tools (TextSeer) has been designed and developed as well as a method to assess strategic alignment. These tools enable a business to better understand their operations. This understanding allows an organisation to manage their processes more efficiently with clear process descriptions and process to strategy mapping functions. Models developed in the study describe process dependencies (e.g. antecedent, sibling), process provisioning, QoS requirements, resource requirements, and the use of other processes.
1.2. Outcomes and Contributions

The following is a list of key tasks completed for this research project.

1. **Development of a strategic modeling language**
   In this task, a strategy modeling language (SML) \[1, 69, 123\] was designed in the form of a modeling notation to aid in identifying and visualizing strategies related to business processes. In addition to this, a strategy ontology \[1\] was created through a review of a large collection of business annual reports and strategy documents. This ontology can be used to describe general organisational strategy that can be then converted into SML for formal analysis. Both the strategy modeling language and the strategy ontology are discussed in Chapter. 05.2

2. **Development of method for evaluating alignment between a set of organisational strategies and a collection of business processes**
   In this task, alignment between organisational strategies and business processes was described through a set of realisation relationships. The result of this task was an application that can show business processes that achieve various organisational strategies. This definition shows the relationship between all strategies and all business processes within an organisation. Each strategy in an organisation is realized by a collection of business processes, and each business process will realize a collection of organisational strategies. Achieved by extending the work on semantic effects of business process models, to demonstrate how an accumulation of semantic effects of business process models will lead to the identification of strategic outcomes. This task is discussed initially in Chapter. 03 and then completed in Chapter. 05

3. **Development of a method for managing organisational compliance.**
   To complete this task, a formal framework for compliance assessment and measurement was developed using an elegant mathematical construct, the ‘c-semiring.’ The result and benefit of achieving this makes it possible to demonstrate the degree of compliance to a set of legislation for any given business process model. This task is described in Chapter. 06

4. **Implementation of a prototype application for automated tool support of the previous tasks**
   In this task, a toolkit was developed for the evaluation of the previous tasks. This takes the form of a programming library called TextSeer. Due to the nature
of work in this area, formal descriptions and definitions can be provided; however, implementation and evaluation are required to show how effective such an alignment between business processes and organisational strategies would be. The library developed is discussed in detail in Chapter. 07.

The contributions of this body of work are as follows:

• A general formal framework for discussing semantic business process composition has been described.

• A method for measuring the distance between process models has been created.

• A framework for measuring imprecise compliance requirements has been proposed.

• A strategy modeling language and a strategy ontology has been created.

• A formal system for measuring alignment has been created and an implemented toolkit satisfying the properties of the system has been developed.

1.3  Thesis Structure

The remainder of this dissertation is divided into five parts; each investigating a different component of the overall strategic alignment framework proposed. Finally after we conclude the case study that formed the basis for assessment of the components of this work is provided in Appendix. A.

1. Part 1 - Chapter. 02 contains a literature review and analysis of existing work in the area.

2. Part 2 - In Chapter. 03 a method for semantic annotation and accumulation is described for business process models. In addition, we provide a method for semantic process composition.

3. Part 3 - In Chapter. 04 we describe a method for conducting semantic business process integration and similarity measurement.

4. Part 4 - In Chapter. 05 a framework for conducting assessment of alignment is described. In addition to this, in this part we also provide details of a strategy modeling language and ontology for writing new strategies.
5. Part 5 - In Chapter 6 we provide a method for measuring compliance over a set of business process models. This compliance framework is an integral part of our overall alignment system.
Chapter 2

Literature Review

It is a mistake to think you can solve any major problems just with potatoes....

DOUGLAS ADAMS

To consider the concept of alignment and what alignment is in the context of this thesis we provide a detailed survey of a collection of seemingly disparate areas. Despite the diversity of topics drawn from, the general concepts and language we use to describe alignment fits under the classification of requirements engineering problems. In this chapter, we begin with a brief introduction to the discrete languages utilized in this thesis and then follow with a survey of goal and strategy modeling. Next we’ll drill down and provide similar descriptions and analysis of verification problems, business process management, compliance, similarity and matching, quality of services and alignment.

2.1 Formal Preliminaries

Throughout the thesis there will be light use of formalisms and logical language. In this subsection, the reader is introduced to some of the most common elements of the formalism used in the rest of the thesis.

In most instances we use a finitely propositional language $\mathcal{L}_{\text{ANG}}$ over a set of propositional letters $\mathcal{A} = \{\alpha, \beta, \gamma\}$, truth-functional connectives $\neg, \land, \lor, \rightarrow, \leftrightarrow$, and the truth-functional constants $\top, \bot$. An interpretation of $\mathcal{L}_{\text{ANG}}$ is a function from $\mathcal{A}$ to $\{T,F\}$ (the domain); s.t. $\Omega$ is the set of interpretations of $\mathcal{L}_{\text{ANG}}$. A model of a well-formed formula $X$ is an interpretation that makes $X$ true. Any sentence in $\mathcal{L}_{\text{ANG}}$
that conforms to the correct syntactic and semantic requirements of a propositional
language is a well-formed formula (wff).

**Well-formed Formula**

1. All atomic wffs are wffs.
2. If α is a wff, so is ¬α
3. If α and β are wffs, so is (α ∧ β)
4. If α and β are wffs, so is (α ∨ β)
5. If α and β are wffs, so is (α → β)
6. No string of symbols is a wff unless it can be derived from 1–5

A set is a collection of objects, each object in the collection is called an element of
the set. Each element of a set is unique and can appear only once. If an element α is
part of a set A, then we denote the membership α ∈ A. If α is the only element of the
set A then A = {α, ∅}. A sequence is an ordered collection of objects. Each object is an
element of the sequence. Sequences are ordered and elements can appear more than
once. A variable is a collection of values, called a domain. The domain of a variable
is a list of all the values that a variable can take, in most of the cases here ⊤, ⊥ is the
domain. Given a set of variables a relation on the set is any subset of the Cartesian
product of their domains.

Sets such as the set of reals, have arithmetic properties such as addition and mul-
tiplication; there is also order theoretic properties defined over an ordering relation
≤. The general properties of an order relation are:

- P1. α ≤ α (reflexivity)
- P2. α ≤ β and β ≤ α implies α = β (antisymmetry)
- P3. α ≤ β and β ≤ γ implies α ≤ γ (transitivity)
- P4. α ≤ β or β ≤ α (linearity)

Order relations that satisfies the properties P1-P3 are called partially ordered re-
lations. Sets with such relations are called partially ordered sets. By developing a
mathematical structure based on partially ordered sets, we can add further abstract
arithmetic properties to non-arithmetic sets. Throughout this thesis we have lever-
aged the mathematical structure of a c-semiring.

A c-semiring \[20\] is a tuple \(\langle A, +, \times, 0, 1 \rangle\) such that:

(i) \(A\) is a set and \(0, 1 \in A\)

(ii) + is called the comparison operation. It is commutative (i.e. \(a + b = b + a\)), and associative (i.e. \(a + (b + c) = (a + b) + c\)), and the identity element for + is 0 (i.e. \(a + 0 = a\)). + is idempotent and gives partial ordering \(\leq\) where \(a \leq_s b\) implies \(a + b = b\).

(iii) \(\times\) is called the combination operation, is an associative operation such that 1 is it’s unit element and 0 is its annihilator (i.e. \(a \times 0 = 0\) and \(a \times 1 = a\)).

(iv) \(\times\) distributes over + (i.e. \(a \times (b + c) = ab + ac\))

A c-semiring has the properties of partial order \[20, 53\] such that Given any c-semiring \(S = \langle A, +, \times, 0, 1 \rangle\), a partial order \(\leq_S\) over \(A\) exists such that \(a \leq_S b\) iff \(a + b = b\). As a c-semiring contains all value combination of an abstract system, all combinations are ordered specifically based on the comparison operator. We generate preference valuation using qualitative measures using the principles shown in \[208\] and decision making using ambiguity response formalisms \[68\]. Within the c-semiring framework \[20\] it is possible to substitute any c-semiring instance into a degree measurement framework.

In classic propositional logic, entailment is a relation between two well-formed formulas \(\alpha\) and \(\beta\). Each formula we will use to demonstrate realization is a prime implicate. Let \(\Pi\) be a theory, \(\alpha\) and \(\beta\) be wffs, then the antecedent formula \(\alpha\) entails the consequent formula \(\beta\) iff \(\alpha \neq \emptyset\) and every model of \(\alpha\) under \(\Pi\) is a model of \(\beta\), denoted \(\Pi, \alpha \models \beta\).

**Tarskian consequence** \(Cn(A)\) is used to denote logical consequences of a set of WFF \(A\), that is given a set of WFF \(A\), what conclusions hold and what can be derived from the conclusions of \(A\). \(Cn(A)\) has the following properties:

- \(A \subseteq Cn(A)\) – **Inclusion**
- If \(A \subseteq B\), then \(Cn(A) \subseteq Cn(B)\) – **Monotony**
- \(Cn(A) \equiv Cn(Cn(A))\) – **Iteration**
2.2 Goal and Strategy Modeling

Consider an example, let \( A = \{ (\alpha), (\alpha \rightarrow \beta) \} \) be a set of WFF. From \( A \) we can derive \( \beta \) holds, therefore \( \{ \beta \} \subseteq Cn(A) \). An observation on Tarskian consequence is that results are infinite. Consider: \( \{ \alpha \} \subseteq \{ \{ \alpha \}, \{ \alpha \}, \{ \alpha \} \} \). The consequence operator is not limited. We introduce a fixed point over the consequence to allow semi-computability.

Given a set \( \alpha \) and a function \( F : \alpha \rightarrow \alpha, x^* \in \alpha \) is a fixed point of \( f \) iff \( f(x^*) = x^* \).

For computability, we require one final property, this is a set minimality property. For any set of closed wffs \( A \subseteq \mathcal{L} \) let \( \Gamma(A) \) be the smallest set obtainable from a closure operation (either \( Cn(A) \) or \( Th(A) \) defined below). This addition weakens the property of monotony usually found in Tarskian consequence.

A graph \( G = \langle V, E \rangle \) is a structure that consists of a set of nodes \( N \) and a set of edges \( E \). Depending on graph type each edge is an ordered/unordered pair of vertices, i.e. \( (n_i, n_j) \) or \( \langle n_i, n_j \rangle \). Two nodes are adjacent when there is an edge connecting them. A path is a sequence of edges \( \langle e_1, e_2, e_3, \ldots \rangle \) such that the last element of \( e_i \) is the first element of \( e_j \).

2.2 Goal and Strategy Modeling

Requirements engineering is a discipline dedicated to the study of the purpose and properties of software and information systems. The resulting artefacts of the requirements engineering process are documents and system designs that benefit the analysis, communication and implementation of information systems. The aim of the requirements engineering process is to minimise incidence of incompleteness (missing from the design), inconsistency (contradictions in design), ambiguous (leading to misinterpretations of the design) [204].

Typical requirements are elicited through various business analysis activities (see below), which result in a set of design artefacts [30]. There are many classes of requirements, which themselves are broken down into other requirement types. Strategies are examples of high level requirements, used by Businesses to guide thinking. Test scripts are examples of low level requirements, used in the development process to guide a particular stage of system implementation. When eliciting and documenting requirements, it is important to document relationships that exist between requirements. These traceability relationships are essential when considering requirements change projects [108].

Formal Tropos is a requirements specification language [62]. It was developed to as-
sist in the verification of requirements and to ensure completeness and consistency of software systems. Further to the development of the language, Fuxman, Liu, Mylopooulos, Pistore, Roveri and Traverso [63] have proposed a framework that support verification of early stage requirements specifications. The tool developed (T-Tool), provides automated support for the verification of requirements written in Formal Tropos [62] by way of model checking.

Requirements completeness has been recently assessed by Cailliau and Lamsweerde in [28] where they have proposed a quantitative risk assessment technique. They have included a probabilistic layer to support behavioural goal characterization with respect to their degree of satisfaction (satisficement).

In Wieringa, Maiden, Mead and Rolland’s study [221] the engineering cycle has been decomposed into a number of business analysis activities. These include problem investigation, solution design, solution validation, solution selection, solution implementation, and, implementation evaluation. They further go on to provide research classifications for requirements engineering work, including: evaluation research, solution proposal, validation research, philosophical papers, opinion papers, and personal experience papers.

When considering the nature and position of the work presented herein, we have taken Wieringa’s classifications into consideration. Through the process of finding results to the proposed thesis and sub-tasks we have undertaken the activities of investigation, design, validation implementation and evaluation. To classify the nature of the work conducted in this thesis, we begin first with a philosophical question and propose various solutions to the questions. The evaluation has been carried out based both formally and through the use of a case study evaluation.

Despite the classifications in [221], we have further invested requirements engineering problems in detail from the perspective of specification and validation problems, business process and workflow requirements engineering, goal and strategy modeling and reasoning, similarity and integration, compliance and verification, and alignment.

To the specification end of the continuum lay modeling languages for strategy and goals. These are used to capture requirements of organisations at a high level and are typically incorporated into long-term planning activities. These goal-oriented description languages are described in detail in the literature of goal-oriented requirements engineering. “A goal is a prescriptive statement of intent whose satisfaction in general requires the cooperation of some of the agents forming the system” [128].

Goal-oriented requirements engineering is described by van Lamsweerde in [201]
as “Goal-oriented requirements engineering is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analysing, negotiating, documenting, and modeling requirements.”. A requirement is a statement of attainment represented as a goal state. A requirement can be explicit functional goals such as make X true or an imprecise software that describes the quality of the system being modeled such as X is maximized.

van Lamsweerde et. al. have extended their work through the KAOS methodology (Knowledge Acquisition in Automated Specification of Software) for describing a formal language and method for requirements elicitation. The KAOS language is used to describe why, who and when to what requirements. The KAOS methodology is useful for populating and decomposing goal graphs during the requirements elicitation phase of engineering processes. In the article [201], the authors, provides a survey of existing goal based requirements engineering frameworks. This work presents an outlook of goal-oriented requirements engineering prior to 2000. Excellent reference for new readers. Looking at the work of [227] the authors describe goal taxonomies, attributes, links, frameworks, specifications, reasoning, verification, validation elaboration, and conflict management using a case study of railways.

The nature of goal driven requirements engineering is discussed in the article [201]. In particular Lamsweerde refines the definition of goals as requirements. Where “a goal is an object the system under consideration should achieve” [201]. the authors go on to describe the process of goal elicitation and modeling, and; then finally moving into the space of reasoning about goals. Then in [202,203], Lamsweerde, discusses a large portion of his previous work. From the division of tasks for initial requirements elicitation to an implementation analysis of KAOS. A number of possible views for operational alignment are proposed (assessing intentions, structure, responsibilities, functions and behaviours). There is an overview of the procedures for creating goal diagrams with different definitions of the breakdown of modeling elements and restrictions on the use of elements. After the introduction to goal models, Lamsweerde suggests that the object models (concepts) be developed in partial conjunction with goal models (the granular details can be inherited from the goal model). This paper a process for progressive refinement, and then introduction of new goals / objects to meet to-be strategies from the as-is goals / objects. Through use of this process, a complete refinement check may be conducted across the as-is, to-be statements. Cardoso et. al. discuss goal elicitation in [30]. In this study Cardoso et. al. explain the process of goal elicitation and then presents a tool that assists in the goal elicitation process. The tool, assists by proposing different softgoals or non-functional requirements previously discovered in other projects and that are stored in goal reference
catalogue. This concept tends to treat goals as being abstracted from a particular system (for the sake of using reference models), which is very different to the approach of Lamsweerde.

The Goal-oriented Requirement Language proposed by Yu in [224] was created to support goal-oriented modeling and reasoning about the nature and meaning of requirements. This language described requirements by relating actors, goals, tasks, and resources through dependencies links. Goal-oriented Requirement Language supports goal-oriented reasoning by establishing correspondences between intentional elements (i.e. goal, softgoal, task, believe, resource) and non-intentional elements which may be imported from an external model, in a scenario. A graphical representation of the concepts in GRL has been introduced by Yu in [224], as the \( i^* \) notation. The notation consists of two model types, a strategic dependency model that models the interaction between various actors within an organisation as well as a strategic rationale, which is used for modeling the decomposition of tasks and goals for specific organisational actors (showing how decomposed tasks contribute to the subgoals of a goal). In the work of Castro et. al. in [32, 205] the Tropos (an agent-oriented software development method based on goal-oriented requirements) methodology for requirements driven methodology for agent development has been proposed. The Tropos methodology considers the entire development process of an agent system using requirements including dependency modeling, rationale modeling, non-functional goal modeling, implementation design and execution. Other goal-oriented requirements engineering methodologies include GOORE [186](a goal-oriented method for requirements elicitation) and Lightswitch [75, 170](definition of early requirements of an enterprise system).

In the InStAl method presented by Thevenet et. al. [190, 192], the strategy of the organisation is represented in terms of strategic objectives and strategic goals with respect to the vision of the organisation’s stakeholder. Each strategy is modeling using a goal-map “A map is an oriented graph where nodes are goals (or intentions) and edges are strategies.” Further work on strategy modeling using intentions has been conducted by Nurcan et. al in [157] where intentions have been used to identify strategies. Strategies are then modeled using goal models through a mapping, where “A map is composed of several sections. A section is an aggregation of two kinds of intentions, source and target, linked together with a strategy.” By modeling strategies using mappings Nurcan et. al. are able to discover strategies and capabilities; however, the authors identify that problems exist in intentions conflicts and identification of high level organisational missions that are not generally related to business processes
2.2. Goal and Strategy Modeling

directly.
The i* [85,152,226,227] modeling framework is popular in the domain of early requirements engineering. Quite a few researches have been conducted based on this framework. Most notably, GRL supports goal-oriented reasoning [54,152] by establishing correspondences between intentional elements (goal, soft goal, task, believe, resource) and non-intentional elements- which may be imported from an external model, in a scenario. In [23] Bolchini has considered the application of requirements management to web applications. To do this, he has extended the i* framework with a hypermedia requirements taxonomy.

Tropos is an agent-oriented software development method based on the i* framework [73,74]. The software system to be developed is analysed with respect to its intended environment including stakeholders and goals. Two models are usually created in this early requirements phase, namely the actor-dependency model and the goal-plan model.

The Lightswitch approach [170,214,215] addresses the definition of early requirements of an enterprise system. Using three main constructs: maintenance goal, achievement goal and belief, the modeller can model the way an enterprise operates in its environment before identifying the goals that the IT systems of the enterprise need to achieve. GOORE is a goal-oriented method for requirements elicitation [186]. This method, accompanied by a computer-aided tool, relies on the domain ontology to help the modeler complete a goal decomposition model.

Strategy formation can be done in a number of ways. Many types of strategies exist [142,143], in early work by Mintzberg et. al. various strategy types and patterns have been identified. Later Mintzberg has described 5P methodology as a series of strategy patterns. The P’s stand for Plan, Ploy, Patterns, Position, and Perspective. In this, the Plan is a consciously intended course of action. A Ploy is a misdirection strategy, created to add diffusion to competitors. A Pattern is a tentative goal awaiting realization through unintended behaviour. Interactions between the organisation and the environment result in the derivation of the organisational Position. Much of the language used for describing strategy in 5P has roots in military strategy where the goal is to deploy resources in a manner to defeat the enemy. Perspective strategizing is the consideration of all elements within the enterprise boundaries; wherein the aim is to prioritize services that have high benefit or value to the organisation..

Yu, Mylopoulos, and Leite have proposed a framework for creating goal aspects with a goal validation framework in [227]. This work provides a formal procedure for decomposing and validating non-functional requirements for source code. This task
is manually conducted by mapping functional goals to non-functional goals in a V-Graph. The procedures that have been proposed could be of good fit for the Strategy-Service Alignment problem between functional goals and service summaries. The drawback of this approach is in the ability to associate compensation for inconsistent NFR’s \cite{212}, or to model across all combinations of functional requirements as in \cite{147,147}.

Work in the area of business strategy as a coalition system \cite{64} has been conducted by Gans et. al. In the work, a methodology for implementation of a coalition system has been shown. A note on this work is that all environmental factors should be considered “partial approaches to evaluating strategies may not only be insufficient but may actually be misleading as prescriptive decision-making tools” \cite{64}.

There have been a number of proposals for the implementation of modeling languages useful for capturing the vision and the requirements of an organisation. Bleistein et. al. suggest that many organisational strategies may be modeled in goal-oriented languages; however, “requirements engineering approaches for organisational IT do not encompass business strategy” \cite{21,22}. Bleistein et. al. have examined the relationship between organisational strategies and systems requirements based on goal modeling, integration analysis (between strategies and requirements using interview techniques), and validation of models (using interview techniques).

2.3 Specification and Verification

A software specification is a description of which the system under scrutiny must conform to. The proposed system must satisfy or satisfice the specification. Requirements engineering is concerned with the production of specifications, including those for implementation, deployment and maintenance of the system \cite{170}. If the system is not implemented correctly then it can have disastrous effects, as shown in \cite{61} which described the impact of the most notorious information system failure in recent times (the London Ambulance Fiasco).

To create reliable systems, a formal methodology or language should be used for the creation of the system requirements. That way, the requirements can be verified using a number of approaches. Verification and specification are important in the space of requirements engineering especially in the realm of business process management. Various schemes for automatic verification of workflow systems have been proposed in \cite{134,144,149,165}. Formal verification in general reduce requirement specifications
to mathematical structures with formal properties that be reasoned over [41].
The most important aspect of verification is the formal definitions given by Letier et.
al. in [127]: Correctness of goal operationalization [127,166] Let a set \( \{R_1, \ldots, R_n\} \) of
required conditions (read as effects) on operations in the operation model correctly
operationalize a goal \( G \) in the goal model, if and only if the following conditions hold:

1. \( R_1, \ldots, R_n \models G \) (completeness)
2. \( R_1, \ldots, R_n \not\models \bot \) (consistency)
3. \( G \models R_1, \ldots, R_n \) (minimality)

In [75] Gonzalez and Diaz reverse the perspectives given by Letier [127] by consid-
ering a process driven requirements verification approach. The aim of which is to
enforce systematic participation in the requirements verification process. Further to
this, in [104], Hummer et. al. have developed a model-driven approach that can be
used to enforce task constraints in business processes.
Montali, Pesic, van der Aalst, Chesani, Mello and Storari have extended this idea
in [144] where they present a framework for specification and verification of cross
organisational requirements. To achieve this, they have used declarative flow def-
initions rather than prescription flow definitions. This is done as they assume that
large and multi-party requirements projects are run continuously and are constantly
refined, which is an accurate assessment of the nature of process systems.
In particular, a number of authors have described the process of goal verification
based on refinement operators. Goal refinement is an interesting area of research
that has been addressed by [43]. In their article, the authors explain general rules
for goal refinement are created with respect to the KAOS language. Further to this
in [108], Jureta et. al. have expanded this idea with the notion of a goal argumentation
method. This means that during the decomposition process that the relation between
antecedent and parent goals can be classified by contributory factors such as support
and justifies. The method proposed insists on the use of adequate documentation of
goal decomposition, so that when goals change, the requirements can be propagated
correctly.
O’Riedl gives an interesting story-based approach to both specification and goal re-
finement activities for requirements engineering in [174]. In this work, he equates
requirements specification to the process of story generation. Where the problem is
to generate fabula (narrative of chronologically ordered events) that meet a set of
given constraints. Much like a sound requirements specification “a sound fabula is one
in which, under the assumption that the world cannot change in unpredictable ways, no event in the fabula violates the ‘physics’ of the story world” [174]. This idea of story-based requirements has previously been addressed by Delgadillo and Gotel in [44], though it is an evolution from agile user-story cards presented by Delgadillo.

In principle, such systematic derivation is hampered by the incomparability in the expressive power of the linear-time and branching-time paradigms. In practice, however, the presented goal assertions are based on formal patterns that hide the differences between the two paradigms.

In [134], Lu et. al. describe workflow verification through the definition of a set of inference rules and workflow constructs. The authors have used planning techniques to show that it is possible to generate viable workflow models that conform to the specifications. As a counter in [213], Weber, Hoffmann and Mendling propose a system for verification of semantically annotated process models that provides a level of verification that is beyond soundness.

## 2.4 Business Process Modeling and Workflow

Business process management has been defined by van der Aalst, et. al. in [2] as “supporting business processes using methods, techniques, and software to design, enact, control, and analyse operational processes involving humans, organisations, applications, documents and other sources of information”. Armistead, Pritchard and Machin in [7] have looked closely at business processes in the management literature, describing business processes as a means for discovering areas of excellence within businesses. Through the investigation, qualities of management such as capability descriptions, knowledge learning and management, market value chain, and organisational coordination have been emphasised as important drivers encouraging organisations to adopt a process-centric operation. Armistead suggests that “processes are a generic factor in all organisations” [7]. Further to a process-centric approach to business management, various languages can be used to describe business processes. Rosemann et. al. have investigated the use of business process modeling and have suggested that “On the one side, it provides a filtering lens that facilitates insights into potential issues with an implemented system. On the other side, it can also contribute to the further development of the selected theoretical basis” [179].

In [196,197], van der Aalst has proposed the use of a petri-net encoding for the modeling of workflow designs. The use of which has spawned an entire area of study into
the properties of wfNets (workflow-nets, a play on petri-nets from which they originate). In addition to this, several tools most notably ARIS [185] have been created that have leveraged this fundamental way of defining workflow systems. The article [228] by Yuan et al. proposes an alternative to van der Aalst’s work on workflow management in particular to the WF-Net. Here a new workflow analysis technique based on petri-nets is proposed and proved to be derivable from a series of workflows. The implication of this work is that the SYNCHRONIZER model item can serve as the foundation of further workflow models. This itself is less of an argument for petri’s nets, and rather an argument away from their deficiencies as the synchronizer takes on the role of JUMPS, SKIPS, Loops, And, XOR and OR splits all at the same time.

Process modeling is not a new area of investigation, with roots in the entity relationship modeling described by Peter Chen in [34]. Models have long been used to incorporate views of the world as a way to analyze specific knowledge. The work of Nicola Guarino in [89] has suggested that any model can be an ontology “as a particular system of categories accounting for a certain vision of the world”. For much of our work in the area of business process management, we model processes to achieve the vision of the organization without irrelevant details. We use a general and industry-preferred notation called business process modeling notation (BPMN) for the representation of business process models.

The Business Process Modeling Notation (BPMN) is a flowcharting technique for creating graphical models of business operation processes, which are generated into Business Process Diagrams (BPD) [220]. The notation consists of a number of graphical elements including activity objects, flow objects, connectivity objects, grouping objects, event objects, annotations and artefacts.

BPMN has been chosen for use in this thesis, as it has been adopted as an industry standard with its uses matching a vast repertoire of workflow patterns [182]. A business process is a series of activities created to fulfill the daily operations of an organization. Each time a business process is executed, it may vary depending on the variables associated with the execution instance. Business processes are complex adaptive systems that have rules placed on them [72]. Each business input should trigger the execution of a new process execution instance. The execution of a new process instance will have a resulting effect (variations of end result) [147]. The instance’s effect can be measured for compliance based on a preference of goodness based on its execution. A business process is a series of activities that are completed by an organizational actor, this includes the control sequence of flow (through deci-
sion points or parallelization of activity flow) and the modeling of messages passed to other actors. A list of core BPMN elements is shown in [220]. These elements have been implemented in a collection of toolkits used for modeling business process models.

Each event icon marks an execution triggering point within a process model; these can be a start, intermediate or end events. Events associated with external environmental changes have some effect on the process. These can be in the form of a signal or triggers such as a received message (email or phone call) or a pre-planned calendar event. During process executions, events take the form of interruptions or exceptions such as in error signals and timers that break an execution after the process exceeds a time limit. For example, given a compliance requirement to complete processing an application in a period. When the time has exceeded the limit, a set of activities to explain the failure of duties must be triggered. Following this, an escalation of activities to a second level operator. End events signal the completion of a process and have the effect of terminating flow and signaling to initiating processes to continue. They may link to a new process that should be executed next or simply terminate. Each event (start, intermediate, and end) may have generic events, timer events, errors, messages, rules, and signals.

An activity is a single step in a complete process. Each activity may itself be a process, however due to the level of abstract being modeled, will only be shown as an activity. Sub-process markers indicate activities that have been defined at a much coarser detail. Each sub-process may be unfolded to display the extra detail or remain hidden to reflect the relevant detail to the model being presented to reduce complexity.

Gateway flow objects direct the path of a business process model. Gateways distribute or restrict flow to a distinct ‘path’ within a process model. Gateways can be of the form: generic, complex, OR, AND, XOR, INC (inclusive).

A detailed description of the business process modeling notation is shown by White in(White, 2006). BPMN has been defined in XML schema’s and a descriptive standard for implementation by the Object Management Group in (Group & Object Management Group, 2011; OMG, 2006). In the work conducted by Ghose et. al. [95] a formal definition of business process models has been provided in the form of a graph-based encoding. Semantic descriptions of tasks are proposed in [95] where the tasks of a business process model, written in BPMN, are annotated with the effects of the tasks. The usage of this annotation allows reasoning over a BPMN model for the detection of inconsistencies and similarity comparisons.

Further to the work of [94,95], the area of understanding process models and quality
of models has been undertaken by Rosemann et. al. in [84, 167, 172, 173, 180]. Through his work Rosemann et. al. [84, 167, 180] have examined the evolution of various process modeling notations, comparing various notations such as Petri nets, ANSI flowcharts, EPC’s and BPMN using quality benchmarks that show elements that are represented in each notation. Similarly Wohed et. al. [222] have examined the uses and practicality of BPMN for modeling Business Processes in [177, 222]. The results of his work show that BPMN is suitable for modeling basic control flow, advanced synchronisation, structural patterns, and multiple instances patterns (that are expected to be found within an organisational setting); however, the business process modeling notation is poorly designed for system design that requires low-level modeling of data, resources, and execution patterns (such as creation, push, pull etc.). In the work of Mendling et. al. in [51, 57, 140], the authors have conducted an empirical surveying to determine the ease of use of BPMN as a notation in contrast to various other modeling techniques identifying various factors that may limit general use of the notation in an organisation. The results of this survey suggest that BPMN is a good notation for the representation of business processes that can be understood by users in an organisational setting.

Service modeling is a refined area of study that examines the atomic implementation of business processes through a service lens. Arsanjani et. al. have described a service “From a business perspective, a service is a well-defined, encapsulated, reusable, business-aligned capability” [8]. Arsanjani et. al. further describe service-oriented modeling and architecture showing the lifecycle of services within organisations. The work in [8] suggests that during process management, when new functionalities for processes are required that a service architecture can leverage reuse of flexibly designed services. The service-oriented architecture presented in [8] is different from an object oriented system design as instead of focusing on program to interface designs, the focus is placed on the general software architecture when designing services that fit across the entire organisation. Further Cherbakov et. al. discuss an outlook for a strategic service implementation that emphasise the use of capability management of cost, scalability, and flexible change for services in [35]. Service description languages such as the Universal Service Description Language (USDL) presented by Kona et. al. in [117] provide a semantic description of services to describe service capabilities at a level of detail aids organisations implement services and describes deployment and composition.

Context awareness in business service management has received some considerable attention recently. Reasons for implementing context aware business services have
been discussed in \cite{164}. In this work Ploesser et. al. suggest that services are not static. Due to changes in the environment a service interruption is likely to occur. In \cite{194,217} Ukor and Carpenter show an abstract method for optimization of service selection. Their work considers all possible execution paths for multiple services and shows how selection can be made based on pre and post conditions.

In \cite{111} Khomyakov and Bider propose a reverse approach to achieving case flexibility by first describing a set of workflow states and then using restrictions of obligations, prohibitions and recommendations a theoretical hybrid automata machinery is able to constructively create all combinations of states and hence describing all that is possible given a set of process and a set of constraints. In \cite{150} Mundbrod et. al. has presented a lifecycle methodology and framework for supporting collaborative knowledge work. Mundbrod has identified a large number of elements that can be used to describe large scale and complex systems including complex financial services and criminal investigation scenarios, which involve highly trained knowledge workers.

Hildebrandt et. al. \cite{92,93} have approached the creation of a dynamic and declarative case management system with a system of Dynamic Condition Response Graphs that provide state transition modeling for case systems. In particular, their work focuses on the execution level store of case models and have provided a rigorous and formal model of case management systems. We believe that their condition response graphs are complementary to our Case Sequence models. Using our framework a case management system can be designed at a broad overviewing level, and then the Dynamic Condition Response Graph can be used to model and assess case behaviour at execution time. In future work, we would like to provide further evaluation of our framework under design usage to compare with the execution support of the Hildebrandt model.

In previous work \cite{24,50,135,175,197}, many researchers have described formal models for graph encoding particular process model types. In our work we provide a general summary for process model formulation (based loosely on the work of our peers); however, we draw attention to the fact that all automation and computation in our framework is done at the design level rather than at the execution level and as such various elements of some graph encodings do not fit with the definitions we’ve provided. By maintaining a process and task definition at the design level, reconfiguration can be computed without the need for execution, so that new adaptive task sequences can be constructed without an example execution trace. For example, in \cite{24} Bose and van der Aalst et. al. have described an execution trace
over the state space of unbounded logs for a workflow net, where their notion of an ordering relation is a sequence of possible state transitions. Our notion differs as it is done at a design level. Further to this, each trace dealt with in this paper through a semantically annotated process model is one of the many possible interleaving execution designs that exists in the process model. In [135,175] Rinderle-Ma et. al. have described a notion of executional event trace that is similar to the executional traces of van der Aalst, this notion of trace like those in [24] requires dropping from design into the domain of execution artifacts.

A key element missing from this story is in how the environmental factor may be used and modeled to show how service selection may be achieved in the face of change. Also, how reparations may be made to compensate non-completed services. In [15,126,128] Letier and van Lamsweerde have attempted to answer this question by developing an algebraic probabilistic scheme that uses objective based mark-up on functional goals, this can be used to determine the degree of satisfaction for completion of a non-functional goal. Letier et. al. provides a propagation method for the decomposition of objectives to further refine goals. The methods described have many benefits for the comparison of similar goal instances, though does not extend to compare the consistency of goals across the entire system design where NFG’s maybe used as qualitative statements to determine the degree of satisfaction for other goals that are raised in [146].

2.5 Compliance

Compliance is a broad and general area that can be considered from the viewpoint of management, legal, and business process across a compendium of disciplines. We consider the characterizations of compliance in [83] to interpret compliance as the adherence of a set of business activities against a set of compliance rules. In this there is a clear distinction between design-time compliance (based on process design and management policies) and run-time compliance (the realization of processes, through use and repetition). Compliance requirements and rules can be broken into contractual rules [83] that can be accumulated across business process models [184]. Once a business has defined its operational domain and a list of potential activities, there should be a level of consistency in the actual completion of processes provided the business logic is correct. There are also times within the industry where some level of failure is acceptable and further work in defining statistical analysis measures
should be undertaken as in [119] to interpret outlier activities and determine a solid variance acceptable for each industry.

“A compliance system is an organisation wide tool that links legislative and business rules to organisation policies and processes. The objective of such a system is to promote a self-sustaining level of operations that minimizes the losses caused to the business through breaches of laws or internal misappropriations” [52].

Compliance is a chain of creation, institutionalization, implementation, evaluation and feedback return [52, 139, 188]. The cost of compliance with various legal requirements such as Section 404 of Sarbanes Oxley is acknowledged with affecting stock prices directly [4]. The punitive measures set in place by Sarbanes Oxley has driven a number of public companies towards privatization rather than continuing in public markets [56].

Motivations for adopting a compliance framework are outlined in [3, 29]. These include: Increasing threats to information security, Alignment of technology projects with strategic goals to ensure maximal value addition to modern product lines, and the increased risks involved with intellectual property for soft service deliveries.

*Meta process statements* define various activities within an organisation. These are generally statements of imperial policies that give an actor a choice of actions by which they can use and combine to complete the task at hand. As various factors affect an organisation and processes change, meta statements must be readily adaptable to meet compliance requirements [46]. Alignment of compliance requirements to meta-process statements is important during an organisation life cycle. Change management and process integration systems can are vital as organisations adapt to moving markets. Each change management scheme should maintain a degree of compliance, this is brought about through minimal change [147].

During a business process, *fulfilment of duties* is the obligations to execute the required duties. In our context, we refer to fulfilment as an agreement of policies that outline methodologies for completing the required tasks. Duty of fulfilment is the product of applying governance policies to meta-process statements within an organisational role [90]. This builds upon the work within the area of contract systems [83].

Compliance regulations and policies define requirements on the business itself. The importance in the creation of *governance policies* is inclusion of penalties for non-compliance [193]. Governance policies can be considered as an authority that provide an actor within the organisation power to perform their duty.

At the most fundamental level, *segregation of duties* means that no individual should have the control of two phases or more than one transaction or of an operation. Crisp
2.5. **Compliance**

Segregation [13] offers the benefit of fraud detection where in most cases the fraud requires collusion by two or more people, and it is much more probable than accidental errors are found [187]. Segregation is employed to ensure that errors and irregularities are prevented or detected during operations by employees (throughout the normal course of the business). Within most multi-agent systems it is a challenge to express and enforce autonomous interactions [66]. There is also currently work in deriving individual activities from collective obligations with deontic logic [37,38].

As standard practice all large sized businesses have some level of auditing providing an abstract understanding of the current business situations ([13]. The major concern with auditing practices tends to be associated with lightened auditing principles. Any system that has a heavy reliance on polite and non-intrusive practices opens itself to hidden fraud [187]. We suggest that auditing and transactional analysis should have a closer proximity to each other or at least a more impartial assessment criteria for auditing practices [66].

**Transactional data and analysis** is of great importance within any organisation, as it allows analysis to be performed to provide precise information about the current operations of the business. These results carry vastly improved confidence in financial reports/ratios, and make progress by reducing the maintenance costs of manual compliance by finding outlier processes to determine common failing points [169]. Automation begins to address the risks of non-segregation of duties by acting as a mandatory buffer between business operations [81].

The degree in which a business process follows its governance policy. This is a performance measure placed on transactional analysis that rates the current operational score against the optimal or best practices ranking provided by the governance policies [81]. Compliance in terms of design is heavily influenced by standards within most corporate governance frameworks. A compliant organisation aims to minimize risks through use of five key principles [16]:

1. **Predictability** - Regulations within a stable system for creating and enforcing policies. If an organisation is consistent with policies that work, then that organisation lowers risks. There is a question of reliability in predictability [119] and propensity to change [47].

2. **Transparency** - Clarity and availability of information. If there exists a clear framework and clearly defined policies then it follows that any person or stakeholder is able to see the chain of reporting, clear rule definitions, and identify accountable parties. We provide a case study to show disparity in providing
2.5. Compliance

information across a population and the importance of transparent communications [25,26].

3. Accountability - Repudiation through segregation of duties is a desirable element in any system. It should be clear that those who control and manage companies must be recognizable for when and how their activities breach predictable rulings [37]. As well as audit trails to lead investigations to guilty parties.

4. Participation - All elements of the principles must be implementable and used. It’s no good if one department adheres to the rules just to find that another does not. In a case study of participation it can been shown that participation can sway overall level of compliance [169].

5. Evaluation - There should be regular evaluations of the policy and the renewal of people’s knowledge of it

Existing methods for assessing and measuring compliance within business operations are largely driven by checklists or transactional monitoring. Balanced scorecards [156] are an organisation assessment device. A business may use a balanced scorecard to rank each process it completes on a matrix to find roughly where it stands compared to the overall business strategy and compare the overall strategy conformance to other businesses in similar industries. Alternatively, standards for policy can be used. Standardized policy creation [3,45,105,106,139] is important in defining compliant operational procedures. This is the creation and adherence to ‘best-practices’ and standards [16,139] involving regular revision of policies and continuous feedback to standards committees.

In [181] Rozinat and Aalst have investigated the conformance of runtime transactions against the designs of the process to be checked. In this work Rozinat et. al. consider both the process sequence fit as well as the structural fit of transaction instances to their design counterparts using metrics. Aalst has continued this work in [198] through definitions of precise translation devices from SOAP messages overlaying a formal Petri-nets to ensure conformance. Similar work has been conducted in [147] through semantic and structural comparisons between BPMN models. Algebraic Frameworks have been defined to act as a translation and reasoning devices for activity level compliance checking against contractual requirements as in [83,193]. Work has been completed in the formalization of contract languages such as FCL [81,83].
Governatori et al. [81,83] provide a reparation chain mechanism for representing statements such as condition $C$ generates an obligation $O_1$, failing which a reparation obligation $O_2$ is generated, failing which a reparation obligation $O_3$ is generated. Clearly a reparation chain can be viewed as an elaboration of an imprecise compliance requirement using a linearly ordered preference structure. The focus of our work is in assessing degree of compliance of process instances with imprecise compliance requirements that have not been elaborated. The use of our technique leads to an incremental elaboration of an imprecise compliance requirement, via the mapping of process instance to a partially-ordered preference structure. There is existing work that seeks to monetize “prescriptive policies” [99], i.e., attach monetary penalties for non-compliance. This article [189] addresses the issue of trust within ebusiness scenario’s providing a framework logic for the description of trusted parties by way of deontic obligations and belief. The framework presented is impractical from an application standpoint; however, a depth of information is provided that is worth extra exploration. Further reading into the logics of this article will be worthwhile.

2.6 Similarity Modeling and Redesign

A sub-discipline of the business process management area is the exploration of alternative process designs. This is studied through the areas of process similarity, process matching and process integration. “Process model matching refers to the creation of correspondences between activities of process models” [33]. There are numerous reasons for this, including the need for integration of processes as a result of restructuring or organisational takeover. In [176], Rosa et al. demonstrate the cost of manual merging showing that manually merging similar processes can be both costly and time consuming. Process integration is an activity where analysts investigate relationships across process repositories to classify and merge similar activities into a standardized system. Integration is the process of merging elements from two similar antecedent processes to create a single process that can be used to replace the original processes. Once similar activities are matched, similar portions of the process can be identified and those segments can be first generalised and then integrated.

Validating quantitative models is of upmost importance when using the models for guiding critical decisions. Before the deployment of the system-to-be, collecting data about the systems allows one to make assumptions on distribution functions of quality variables, validate the refinement equations used in the model, and possibly
rectify them. In contrast with quantitative techniques based on subjective criteria, our approach makes it possible to validate or invalidate the models. There are three general activities for evaluation of alternative system designs. For each alternative activity compute the objective functions of the higher-level goals by bottom-up propagation from the estimated distribution functions. For each alternative model, the goals achieved by each model must at the very least realize the goals of the antecedent process. Compare and evaluate differences in the syntactic structure of the models. Compare and evaluate differences in the semantic meaning of the models [147].

In [176], Rosa et. al. attempt to formally define process matching as a mapping. Given two process models $G$ and $H$. Business process matching is the procedure of finding a partial injective mapping $M$ of nodes in process $G$ to nodes in process $H$, for which some real function score is maximal. This definition is based on the original graph edit distance work of Bunke. Alternatively [51] “assume two process models $p_i$ and $p_j$ to be similar, if they expose a common share of behavior”.

Business process matching is the procedure of finding a partial injective mapping $M$ of nodes $n_i \in p_i$ to nodes $n_j \in p_j$ such that some real functional score is maximal [125,176].

Approaches to similarity and matching include work on edit distance, particularly hamming distance, Levenshtein, and Damerau–Levenshtein measures. Hamming distance is a measure on two arrays of the same size. The distance between two arrays with Hamming is a count of the number of non-equivalent characters at each place in the string array. This can be viewed as a substitution operation.

String 1 – EVAN
String 2 – IVAN

Using the Hamming Distance, there is a distance of one character between the two strings. Levenshtein Distance extends the definition to work on difference length string arrays, by including substitute, delete and insert operations.

For example, given the strings:
String 1 – EVAN
String 2 – IVANIA

The first character of string 2 must be substituted from an I to an E. Then there are two delete operations that need to be performed, where the two characters IA must be deleted from the end of String 2 to make it equivalent to String 1. Finally the Damerau–Levenshtein distance adds a transposition operator for any two adjacent characters in a string array.

Given the strings
2.6. Similarity Modeling and Redesign

String 1 – EVAN
String 2 – NAVE

Under Levenshtein, minimally four substitution operations need to be performed. Using Damerau–Levenshtein, the first and last characters of String 2 need to be substituted; however, the transposition operator can change the positions of the second and third characters, making the distance between the two strings 3.

The edit distances shown above are usually used when performing string similarity or some measure on the similarity of activities in a model. When considering similarity of graphs there are graph edit distance algorithms and isomorphism measures.

Given two graphs \( g_i, g_j \), a graph isomorphism between the two is a bijective mapping of nodes where

- Every node in \( g_i \) has an equivalent node in \( g_j \)
- Every edge between two nodes in \( g_i \) has an equivalent edge in \( g_j \)

Jaccard’s Similarity Coefficient can be used as a fast measure of graph similarity as well. It is typically used for comparing the similarity of sets.

\[
J(A, B) = \frac{(A \cap B)}{(A \cup B)}
\]

In our chapter on process integration we demonstrate a syntactic measure based off this similarity coefficient.

Pebble games are and pebble game algorithms provide a fast and efficient way to find graph isomorphism. Pebble games are two player games played by Spoilers and Duplicators. Played on two graphs. Spoilers try to prove that the graphs are different. Duplicators try to pretend that they are really the same. The game is played in rounds. At the \( i^{th} \) round, the spoiler chooses one of the two graphs and one of the vertices of that graph (\( a_i \)). The duplicator must respond with vertices of the other graph (\( b_i \)). If, after \( p \) rounds, the mapping from \( a_i \rightarrow b_i \) is not a partial isomorphism, the spoiler has won.

In the space of state machine similarity, bisimilarity is a generally accepted way to measure similarity. Bisimilarity is a recursive similarity measure that can be defined as being either forward or backwards. Two states are forward bisimilar, if they start at a bisimilar state and there is a transitional trace that results in a bisimilar end state, where each state in each trace is bisimilar. The reverse holds for backwards bisimilar states. Used in [153,183].
2.6. Similarity Modeling and Redesign

Remco Dijkman, Dumas, van Dongen, Kaarik, & Mendling, [49] & M. Kunze, M. Weidlich, and M. Weske [121] have implemented a cut down version of bisimulation in their work on behavioural net similarity. To create a behavioural net first break process models into behaviour matrices. For each process trace, there are either strict or weak order relations between nodes. Similarity is defined as the difference between two matrices with respect to various relations (using Jaccard coefficient).

Mansar and Reijer’s article [137] is of interest as it examines driving forces and leading factors that contribute to an organisation change management operations. Mansar and Reijers have in other work [154] identified a collection of successful process design qualities. In this article, they have present recent work in the form of a survey of industry leaders in change management across the UK and Netherlands. They refine their previous lift of important facts down to a list of 10 driving forces and focal areas that need to be addressed when conducting any change management exercise. This includes task elimination, integral business technologies, task composition, parallelism, specialist-generalist, resequencing, integration, empower, numerical involvement, order assignments [137].

In [153,183], Nejati et. al. Describe the general approach to statechart matching and merging. The authors start with eCharts, a formal statechart model, similar to a graph encoding of a process model. They then describe the process for both static and behavioural matching. In static matching pairs of node/edges are matched to one another using a string similarity algorithm. They then go on to describe behavioural similarity using state-machine bisimilarity descriptions. Bisimilarity is a recursive similarity measure that can be defined as being either forward or backwards. Two states are forward bisimilar, if they start at a bisimilar state and there is a transitional trace that results in a bisimilar end state, where each state in each trace is bisimilar. The reverse holds for backwards bisimilar states.

Weidlich, Dijkman and Mendling published a general model for process similarity in [218]. In which a great deal of work has been describing and breaking down the elements needed for matching and similarity. Starting with activity matching, they describe the notion of matching at a syntactic and semantic level. They also raise the idea of abstraction based similarity matching, where a single activity in one process may correspond to several activities in another process. They then provide a means to conduct fragment based matching (matching small parts of the process). Matches found in this step would be useful in the abstraction modeling space. Following this Dijkman et. al. have furthered this work and proposed a metric for conducting an assessment of process model similarity in [49], in this work they break process
models into causal graphs which are sequences of tasks that demonstrate *behaviours* within a process model (the similarity measure is a partial bisimilarity search). From this point, they produce a measure of node similarity (number of similar nodes in a causal graph), and structural similarity (graph edit distance discussed in [18]). Their contribution is the use of causal similarity measures, which are computed on a subset of behavioural profiles of a process model. When comparing the similarity of task names, Dijkman et. al. suggest that implementation of their metric may consider the use of syntactic, semantic, attribute, type and contextual similarity functions. The fundamental problems that arise from this work are the misunderstanding of the computability of similarity of LTS equivalence in small process models. In addition to this, the authors do not demonstrate the appropriate properties of metrics for their similarity measure.

The above methods for comparing syntactic similarity have been used to find adequate similarity between graphs where there is a clear correspondence between the activity labels on nodes. The challenge of similarity and matching is much harder when node labels are not the same. Approaches to this area to date include language and passage based similarity measures. In our chapter on process integration and similarity, we propose a novel approach to this problem. [219] adopt a passage-based approach to similarity matching. Firstly, they transform a workflow model into a refined process structure tree. Then, they break processes into passages of text (sequences of tasks form passages). They then compute the probability of terms distribution (what is the likelihood of a particular description being used at a point in a process model). They then use the Jensen-Shannon Divergence to measure the similarity between two passage distributions. The result is a measure of similarity between two process models.

Klinkmuller, Weber, Mendling, Leopold, and Ludwig [112] present an alternative matching algorithm that applies more rigor to the label matching. They start with activity matching based on bag-of-word’s similarity. Each task label is tokenized and then each word in a task label is matched against other words in the label being compared against. They extend this with a word pruning algorithm that summarizes long labels.
2.7 Quality of Service

When designing value metrics there are both quantitative measures (fixed real values) along with qualitative measures\(^{[208]}\). Using in decision making frameworks \(^{[68]}\) we may show disambiguation in non-crisp quality of service metrics. To achieve this we must show disambiguation in the definition of preference values. We use a mathematical structure, the c-semiring to do this.

**Definition 1: c-semiring \(^{[20]}\)**

A c-semiring is a tuple \(\langle A, +, \times, 0, 1 \rangle\) such that:

(i) \(A\) is a set and \(0, 1 \in A\)

(ii) \(+\) is called the comparison operation. It is commutative (i.e. \(a + b = b + a\)), and associative (i.e. \(a + (b + c) = (a + b) + c\)), and the identity element for \(+\) is \(0\) (i.e. \(a + 0 = a\)). \(+\) is idempotent and gives partial ordering \(\leq\) where \(a \leq_S b\) implies \(a + b = b\).

(iii) \(\times\) is called the combination operation, is an associative operation such that \(1\) is its unit element and \(0\) is its annihilator (i.e. \(a \times 0 = 0\) and \(a \times 1 = a\)).

(iv) \(\times\) distributes over \(+\) (i.e. \(a \times (b + c) = ab + ac\))

\(\square\)

A c-semiring has the properties of partial order \(^{[20, 53]}\) such that Given any c-semiring \(S = \langle A, +, \times, 0, 1 \rangle\), a partial order \(\leq_S\) over \(A\) exists such that \(a \leq_S b\) iff \(a + b = b\). As a c-semiring contains all value combination of a compliance system, all combinations are ordered specifically based on the comparison operator. This is a trait of c-semirings that allows us to rank and rate various levels of compliance. We generate preference valuation using qualitative measures (as in the previous section) using the principles shown in \(^{[208]}\) and decision making using ambiguity response formalisms \(^{[68]}\). This has been shown in the previous section. Within the c-semiring framework \(^{[20]}\) it is possible to substitute any c-semiring instance into a degree measurement framework.

We model the quality aspects of a BPMN model using an algebraic scheme developed within the constraint modeling literature \(^{[132]}\). We define the quality of service metrics using constraint semirings (c-semiring) similar to \(^{[97]}\) we embed c-semiring values into process statements as ranking scales for singular requirements.
Due to the nature of c-semirings, we can formalize multi-criteria optimization problems as one may encounter in a business domain (e.g. Time vs. Money). A c-semiring is a tuple $S = \langle A, +, \times, 0, 1 \rangle$. When considering multiple QoS requirements a combination of multiple c-semirings can be used to produce orderings over the domain. Using qualitative measures (formalized in [208]) in decision making frameworks [68] we may show disambiguity in non-crisp quality of service metrics. Such that a “good security measure” can be combined with a “fast product delivery timeline” to be better than a “bad security measure” and “fast product delivery timeline”. 

$\langle \text{Security}_{\text{Good}}, \text{Delivery}_{\text{Fast}} \rangle \leq \langle \text{Security}_{\text{Bad}}, \text{Delivery}_{\text{Fast}} \rangle$ where Good $\triangleright^\ast$ Bad (Good is unambiguously better than Bad in every probability scenario). This can be considered an abstract measurement between two ranked but incomparable elements as in [70].

There are systems already defined in the area of penalty addition to behaviour patterns [223] [158] as well as cost-benefit analysis systems based on semantic QOS frameworks [65]. The framework we have chosen is described in [132] where each QoS ranking scheme is placed into a constraint semiring. C-semirings have been used to formalize soft constraint problems [97] and [107] where different tuples in a constraint satisfy the constraint to varying degrees.

## 2.8 Alignment

Combining the approaches to business process management with the methodologies and formal description of strategic modeling through goal-oriented engineering we have shown how to describe the strategy and the relation between strategies and business processes. To review alignment, one must view literature in the space of requirements satisfaction, specification consistency and verification as well as interoperability.

Peter Wegner has described Interoperability as “the ability of two or more software components to cooperate despite differences in language, interface, and execution platform” [216]. Further to this, interoperability is presented by Lea Kutvonen in [122] as a means to provide agile, loosely-coupled services across organisational boundaries.

In [122], Kutvonen has reviewed the use of capabilities with the mutual communication of information, proposals and commitments. The contribution of this work has been to identify pain points in the lack of standardisation and the need for service oriented architecture in business settings. In [17], Beydogan argues a similar point,
though from an industry specific perspective (namely telecoms). Beydogan does indicate that the benefit of standards is the adoption and use by the marketplace. Baldoni, Baroglio, Chopra, Desai, Patti, and Singh [12], have reviewed this concept in the space of conformance to interaction protocols in an agent setting. The result of which is a formalisation of interoperability and conformance. Their formalisation is based on the existence of a relation between two parties, where each party is compatible with the other. They further describe an alignment between parties as a relation where both parties are mutually compatible.

Zirpins et. al. [231] have described the alignment of processes to services using capabilities and role based relationships. The work in [231] provides an excellent service adaptation environment that could be leveraged with this work and work in [211] to describe a capability based change management framework.

Digging deeper into the alignment and decision support systems space, [136] Ma, Lu and Zhang have presented a logic based approach to multiparty/criteria decision making (MCDM) processes as well as an accompanying tool (Decider). This work considers the use of a rank knowledge system based on the aggregation of subjective (non-fact) and objective (sensor reading fact) knowledge. The importance of considering MCDM with alignment is the process and step taken to reach a conclusion.

**Step 1**: identify alternatives.
**Step 2**: identify hierarchies of criteria and evaluators, as well as their weights.
**Step 3**: identify information sources and their connection with criteria.
**Step 4**: collect information from information sources.
**Step 5**: evaluators evaluate collected information to generate initial decision matrix for each alternative.
**Step 6**: apply the fuzzification method to assessments in initial decision matrix.
**Step 7**: apply the fuzzy aggregation method to obtain an overall assessment on each alternative.
**Step 8**: generate ranking for each alternative by the fuzzy aggregation method and ranking strategy.

These steps appear generally in the space of interoperability, as well as in the management literature. With respect to the notion of alignment in management, Wang and Ghose in [210] have proposed that strategic alignment is generally defined between two or more strategies, “basic alignment between a pair strategies holds in situations where there are no impediments to the concurrent deployment of both strategies”. Which echo the notion of conformance and consistency. The approach offered by Wang and Ghose [210,211] focuses heavily on strategy to strategy alignment with varying de-
2.8. Alignment

degrees of strategy operationalism described through resource deployment and strategic effects showing the possible types of inconsistencies that may arise across strategic goals. The crisp description of strategic modeling languages described in [211] highlight flaws in the strategic views taken in [21].

In the article [54], Edirisuriya and Johannesson, have described a methodology and framework for the alignment of value models to process models by translating from $e^3$ value models into activity dependency models and then again into business process models. This work is not grounded by any clear evaluation; however, does offer a value based understanding of abstracted strategy alignment from $e^3$ to BPMN.

The article [160] presents a framework and ontology for interoperability in e-business models. The framework examines a number of strategy, e-business, architectural, and other models. Where-as the ontology considers value propagation from infrastructure to the customer. In [76] Gordijn, expands on the idea of an e-business ontology and presents his $e^3$ value model. The $e^3$ value model examines three perspectives of an organisation, one of which is focused on business strategy modeling. The other forces in place on the organisation are also modeled (these include the value creation perspective and the IT architecture perspective). The use of separation of concerns in Pijper’s work aids in clarifying discussions between relevant stakeholders. $e^3$ is used to model the value proposition of activities in the business, modeling includes meta-model structures and the relation and use of Use Case Maps within an $e^3$ value constellation. Throughout his work [5,11,76,80,109,161,162] Gordijn has created a complete ontology for providing descriptions of various value modeling including syntactic restrictions on the use of linking between actors and activities. The work on $e^3$ value models is focused mainly in the space of knowledge representation and little on reasoning about action.

In the space of reasoning about action, Koliadis et. al. [116] have proposed a framework for aligning business processes to service capabilities. The framework uses semantic effect accumulations over BPMN models to describe relationships mapping effect scenarios to service outcomes. The framework differs from that presented in this thesis not only through much more detailed and extensible formal descriptions (presented in this thesis), but also in that we use the strategy modeling language as a basis for goal relations. In addition, there is no use of QoS measures during the realisation process in the work of Koliadis et. al. The precursor to [116] is described in [114] where Koliadis and Ghose introduce the notion of relating goals (functional goals - from an and/or decomposition tree) with the accumulated effects of processes. This article describes the fundamental relationship between goals and effects show-
2.8. Alignment

ing how processes are related to requirements. Secondly, the article introduces satisfaction goals by the semantic effects of the process. Satisfaction is based on the relationship between process trajectories (or scenario pathways).

In the wider spectrum of methods for relating strategic level goals to business processes, Anton [6] has described an approach to process alignment through a series of model transformations. The primary focus of Anton’s work is on goals and the analysis of what role they play within an organisation. In [6], strategy or high level goals are refined to operational goals and are then typed using general goal subdivisions like maintain, achieve, etc. The article introduces a basic notion of constraint satisfaction and process activity ordering for goal plan realization. This work is still in its early phases and does not distinguish between process activities, activity goals, and goal refinements of strategic goals.

In [30], Cardoso et. al. have shown a method for eliciting non-functional goals from business processes (with a practical case study in a Brazilian hospital). The authors provide a method for the construction of goal decomposition trees, and then provide a method for composing multiple trees to describe organisational strategy. For both Cardoso and Anton, the work appears to be lacking descriptive details beyond a methodology for constructing candidate models of business/strategy relationships. Neither framework has a method for assessing the correctness of models constructed with their implementations.

From a more general view, there have been some modeling languages used to capture the vision and the requirements of an organisation. Of these modeling languages, only a handful attempt to decompose goals to an activity level. Of these, all identify a point in the decomposition chain where descriptions go from goals to activities as being strategic alignment. Closest to this is Bleistein et. al. [21][22][206], who have examined the relationship between organisational strategies and systems requirements based on goal modeling, integration analysis (between strategies and requirements using interview techniques), and validation of models (using interview techniques). Their approach is a comprehensive vision of strategic alignment in the form of goal decomposition. The framework they have presented lacks in-depth definitions of soft goals and optimization objective alignment. Additionally it does not take into consideration the potential corruption of a global solution by task to goal mapping at a local level. Further to this, the case study in the article appears to require a high degree of human manipulation and description. In our implementation section the basic form of consistency checking required to complete simple goal/activity consistency checking as described in Bleistein’s framework is overwhelmingly large, making the
approach taken by Bleistein infeasible at an enterprise level without automated processing.

Koliadis et. al. [116] have proposed a framework for aligning business processes to services capabilities. The framework uses semantic effect accumulations over BPMN models to describe relationships mapping effect scenarios to service outcomes. Our framework differs from the framework for alignment in [116] not only through much more detailed and extensible formal descriptions, but also in that we use the strategy modeling language as a basis for goal relations and we also consider ranked realization. The precursor to [116] is described in [115] where Koliadis and Ghose introduce the notion of relating goals (functional goals - from an and/or decomposition tree) with the accumulated effects of processes. This article describes the fundamental relationship between goals and effects showing how processes are related to requirements. Secondly, the article introduces satisfaction goals by the semantic effects of a process. Satisfaction is based on the relationship between process trajectories (or scenario pathways).

Zirpins et. al. [231] have described the alignment of processes to services using capabilities and role based relationships. The work in [231] provides an excellent service adaptation environment that could be leveraged with this work and work in [211] to describe a capability based change management framework.

In the wider spectrum of methods for relating strategic level goals to business processes, Anton [6] has described an approach to process alignment to $e^3$ value models through a series of model transformations. The primary focus of Anton’s work is on goals and the analysis of what role they play within an organisation. In [6], strategy or high level goals are refined to operational goals and are then typed using general goal subdivisions like maintain, achieve, etc. The article introduces a basic notion of constraint satisfaction and process activity ordering for goal plan realization. This work is still in its early phases and does not distinguish between process activities, activity goals, and goal refinements of strategic goals.

In [30], Cardoso et. al. have shown a method for eliciting non-functional goals from business processes (with a practical case study in a Brazilian hospital). The authors provide a method for the construction of goal decomposition trees, and then provide a method for composing multiple trees to describe organisational strategy.

For both [6,30], the work appears to be lacking descriptive details beyond a methodology for constructing candidate models of business/strategy relationships. Neither framework has a method for assessing the correctness of models constructed with
A framework for goal operationalism has been rigorously constructed by Leiter, Ponsard et al. [127, 166] showing a crisp goal satisfaction framework that can be used to describe the satisfaction of software systems over time.

Pijpers et al. have presented a framework and methodology for modeling business strategies called the \( e^3 \) force in [162]. The \( e^3 \) force examines three perspectives of an organisation, one of which is focused on business strategy modeling. The other forces in place on an organisation are also modeled (these include the value creation perspective and the IT architecture perspective). The use of separation of concerns in Pijper’s work aids in clarifying discussions between relevant stakeholders.

Through the literature reviewed, it has become abundantly clear that there needs to be a link between business process models and strategies. The work that is presented in this article provides the next logical and innovative development towards a formalization of the relationships that should exist in any general SOA framework.

2.9 Summary

The work in this dissertation draws inspiration from and provides a contribution to the areas of process similarity, compliance, and alignment; with an aim to bring these separate areas together. In this chapter, we have provided a comprehensive review of the landscape of these topics along with the identification of key researchers in each space. In summarizing the state of the art and analysing the various gaps in the knowledge that exist across the spectrum of requirements engineering we have motivated and described where the contributions originating in this thesis fits in the broader research context.
In this chapter, the set of languages used to describe process models and strategies are introduced. This includes a process modeling notation used to provide abstract concept over the operational domain. In the context of this dissertation, a process model can be viewed as a representation of a set of activities, and decision rules. The purpose of process modeling is to show the activities that actors or systems perform. Each process should represent a specific, repeatable set of steps. Process models are created from a combination of activities, decisions, and sequence flow. An activity is a single unit of operation and describes a specific step that can be completed. A decision is a point where the model splits. Depending on the environmental state or a user choice, only one path is chosen to follow during the execution of a process.
Sequence flow is the transition between two activities. Sequence gives the order to the activities and decisions in a process. In addition to a tradition process model, we incorporate a semantic description layer to our processes. This semantic layer gives rise to richer analysis on the model in a context of a given business.

In previous work \cite{196,197}, Petri-nets have been used to represent process models. Petri-nets are a graphical notation made of a number of elements that can be used to visualize and analyze systems in a formal manner. A Petri-net can be represented as a graph, with two types of nodes. The node types are transitions and places. Places are states and transitions are representations of state change. The formalism of a Petri-nets in graph representation has been provided by van der Aalst in \cite{197}.

**Definition 2: WF-Net \cite{197}**

Let $N = \langle P, T, F, l \rangle$ be a WF-Net where:

- $P$ is a set of places, $T$ is a set of transitions, $F$ is a set of edges, and $l$ maps transitions to labels.
- $P \cap T = \emptyset$
- $F \subseteq (P \times T) \cup (T \times P)$
- $l : T \rightarrow L$
- There is exactly one $i \in P$ s.t. $\bullet i = \emptyset$
- There is exactly one $j \in P$ s.t. $j \bullet = \emptyset$
- For all nodes $\in P \cup T$, nodes is reachable from $i$ and $j$ is reachable from nodes

The use of petri-nets and other means to describe process models for verification with business users has been notoriously difficult to do \cite{151} and there have been many studies into the usability of various process design notations \cite{72,118}. The Business Process Modeling Notation, BPMN \cite{50}, is a standardized notation for creating process models, used to represent business specific process models. It is formed using a collection of activities, gateways, events, sequence flows, pools, swim lanes, and message flows. In this thesis, we will use BPMN to represent process models. A set of business process models is refered to as a process portfolio. In much the same way as an investment portfolio, a process portfolio is representative of an organisations process assets based on their existing needs and functional requirements.
3.1 Semantic Processes

A process model is typically a representation of the systematic activities that need to be performed in a given order to archive some goal (instructions or how definition). Process semantics provide a declarative specification of a process design; that is, what the process is seeking to achieve. Previous work in this area [95] has described a method for semantic annotation of business processes. This is an effective way of adding semantic descriptions to process models as it produces reusable artefacts (annotated activities) that can be reasoned over. To construct semantically annotated business process models, analysts annotate activities in the model with descriptions of the changes that occur as a result of the activities execution. Such results are referred to as effects of an activity. For each activity or event, state semantics can be annotated. These describe state changes on reaching the activity, we refer to these annotations as immediate effects. Given a set of WFF $A$ describing the state at activity $a_i$ and a set of WFF $B$ describing the state at place $a_j$, if there is a sequence flow and transition from $a_i$ to $a_j$, then the state of the system has changed. As an example, consider $A = \{ \text{LightOn}, \neg \text{LightOff}, \text{Raining} \}$, and $B = \{ \neg \text{LightOn}, \text{LightOff}, \text{Raining} \}$, and let a knowledge base $KB = \{ \text{LightOn} \rightarrow \neg \text{LightOff}, \text{LightOff} \rightarrow \neg \text{LightOn} \}$. With the knowledge base $KB$, it is safe to reason that when the state changes from $a_i$ to $a_j$ that the light goes from being on to being off, and it continues to rain outside. The reasoning takes place and leverages the supplied knowledge base. This is adequate for small state machines, though when working on more complex problems defining a knowledge base for all possible state changes is not practical. One way to address this issue is the use of the closed world assumption, that is for anything that cannot be shown to be true is false and that unless specified all objects in the world remain static. This means that we can reduce our description of states to be $A = \{ \text{LightOn}, \text{Raining} \}$, and $B = \{ \text{LightOff} \}$. We use annotated effect scenarios to provide semantics for processes. For a more complex example (see Figure 3.1), an activity Check employee database for suitable replacement within a human resources process model could have the immediate effect: ConfirmedEligibility. Similarly, the event no suitable replacement found has an immediate effect: $\neg$HolidayProvisioned.

An effect scenario $e$ describes one possible state of affairs that may exist after executing an instance of an activity or part of a process model. The effect scenarios of activities in a process model are semantic annotations that describe the resulting changes of state brought about by executing the activity and its antecedents. By reasoning with process effects, we are able to capture the organisational operation
model, i.e., “what does this process do?” This is important as it allows us understand what happens as a result of a business process execution; and what execution
3.2. Need for Semantic Process Composition

Research in business process management (BPM) is focused on the activities of organisations. One crucial research challenge in the space is that of activity fulfilment in dynamic environments, fulfilment refers to the execution of the activity such that there is partial goal satisfaction. There is an increasing need for case handling and management as an alternative to traditional straight-through processing. Straight-through processing deals with the construction and operation of repeatable workflow scenarios a process designer has created for the organisation. In other approaches that rely on syntactical process analysis, no information as to what processes do can be extracted from the process models except for the names of the tasks. This makes pure syntactic analysis pointless when attempting to answer “what?” questions about process models.

A QoS annotation $\upsilon$ can be used to describe the service characteristics of particular activities in a process model, for example the cost of an activity or the activity runtime. Usually these annotations represent previous runtime information about an activity, such as the time it should take to execute a particular task, or the cost estimation for utilising some resource. They are and should be included during the design time documentation of a process as they can help to demonstrate end-to-end process qualities (such as overall average cost for running an end-to-end process). One way to achieve this is to update QoS annotations at the end of an audit period. We leverage and build on the work of Hoesch-Klohe’s et. al., framework Abnoba [98], which can be used to annotate QoS capabilities $\upsilon$ to activities in a process model. These activities can then be pairwise accumulated, to show capabilities for the process.

Through a mechanism of accumulation of effect scenarios and immediate effects, the effects of a process model can be found. Koliadis et. al. [71], has described a general function for accumulation that takes two immediate effects and returns a consistent effect scenario that is the effect scenario. This accumulation can be done in a pairwise manner across a process model to find the effect scenarios of the process. In original work by Koliadis et. al. the authors left some confusion over the use of pairwise accumulation and the statespace, with respect to continuous pairwise accumulation of effect scenarios and immediate effects. In section §3.4.1 we describe an updated accumulation function that clearly describes how to conduct process accumulation.
and process designs, whereas, case management investigates roles, life cycle, and activity implementation from a more consumer or interactional point of view \([135]\). A key difference between prescriptive processes and declarative process is the definition of control flow. A prescriptive process is invariant to change; however, declarative processes may be modified during execution. Repeatable and straight-through processes allow easy modeling and optimisation of basic activity based value chains \([148]\) and these are classic examples of prescriptive process designs. Poor outcomes typically result from the application of prescriptive process modeling to knowledge intensive activities \([230]\). A declarative semantic process composition approach to process management makes it possible to create knowledge intensive workflows that are not possible to model using traditional BPM methods.

“Case management is built around the concept of processing a case, a collection of information and coordinated activities, by organisational knowledge workers” \([230]\). Typically, a case is a focused view of an interaction with a business unit or organisation by an external entity (customer). A customer, driven by some desire or need, engages with an organisation. These engagements typically result in mutual exchange for services and resources. Through these interactions, various processes compositions and choreographies create a semi-coherent procedure aimed at satisfying the customer’s primary goals or desires \([230]\). This differs from traditional workflows, which make personalised customer transactions and narrative based progressions impossible. Prescriptive processes also typically mean that a client engaging at multiple touch points will need to repeat activities, such as explaining goals several times for each process context. Within a case management framework, a customer engagement case contains all details of the customer’s goals and past interactions. As such, all relevant data and information are available from within the processes and also for the composition engine, making service much more personalised \([230]\).

During the creation of workflow systems, process designers strive to create process models and designs that benefit many varying use cases \([21,54,211]\). The problem for these activities is defining processes that can be used in varying contexts based on customer demands and intentions \([80,116,211]\). Being able to dynamically construct a process that effectively works to satisfy consumer goals as well as business goals taking into consideration operational and historic context is necessary in a business setting \([80,115,116]\).

In this chapter, we describe the use of semantic process management to model task composition that provides declarative support to organisations.
3.3 Motivating Example

Throughout this chapter, we will use a motivating example for pension administration company PAC call centre described more thoroughly in Appendix A to introduce the conceptual building blocks and finally a framework of dynamic and declarative process composition. The knowledge systems used in this chapter are the strategic landscape, process repository, business rules and knowledge base. The strategic landscape includes details of the rationale (described in Appendix A) and desires that PAC wishes to achieve with its call centre. A process repository is used to maintain all operational activities that PAC is capable of performing. Business rules are integrated into processes to maintain application logic across the operational elements of PAC. A knowledge base of key definition and logical correlations is used to support decision making between strategic and operational levels.

**Process Repository** The example case for the company PAC described here is of a complaint handling process. The complaint handling process is a common process that is deployed in most customer call centres. The process has been depicted in Figure 3.2.

The process flows as follows: if a customer is upset at any point in time with the level of service or products that they have received through a transaction with PAC, they will call the call centre and make a complaint. The first stage of a complaint handling process is to ensure that the customer details are recorded, and as such verifying customer details is the first task completed by the call centre agent. In the event that a customer cannot be verified over the phone they will be directed to make their complaint in person at the closest local store or business kiosk. This is marked in the model as an orange task, as it is an exceptional task.

**Business Rules** After a customer has been verified, the complaint is recorded and assessed against a collection of business rules. Business rules can be represented in any number of notations, including SBVR (Semantics of Business Vocabulary and Business Rules), plain text and processable rules formats (for example drools rules language). To avoid confusion we are using a Drools spreadsheet encoding of a set of simple decision rules. These are shown in Figure 3.3. The first section of the rules file describes the rule namespace and reference classes, the second section describes a set of mapping from state to conditions for the process flow. The rules state that if a complaint is made about long waits, that the process should be directed to the close case task, otherwise if the customer is experiencing call drop outs that their case should be escalated for further customer satisfaction management.
When the process has been completed, either the customers will have been directed to a business centre / kiosk, the customers complaint will have been recorded.
3.3. Motivating Example

Figure 3.3: Decision rules for PAC complaints handling process

or the customers complaint will have been escalated for further processing by a customer engagement officer.

**Strategic Landscape** A strategic landscape is a list of strategic goals that describe values that the call centre wants to realize. We have used an i* goal notation to model strategies [225]. A goal model, describes the organisational desires and intentions. A goal model describes the *why* of operations. Hard goals (rounded ellipses) are used to describe functional goals, and soft goals (cloud shaped) are used to describe optimisation objectives. In Figure 3.3, a goal model of the call centre has been provided and describes the heirarchy of soft / hard goals for the call centre.

Each functional goal usually has an included description that can help during strategic alignment. The strategies of the PAC call centre are as follows:

- (Optimisation) Minimize call centre staffing costs
- (Optimisation) Minimize complaint escalations
- (Optimisation) Minimize trouble report rates
- (Optimisation) Maximize trouble tickets cleared
- (Optimisation) Minimize number of open trouble tickets
- (Optimisation) Minimize trouble ticket volume
3.3. Motivating Example

Figure 3.4: Goal model for PAC Call Centre

- (Optimisation) Maximize number of trouble tickets closed
- (Optimisation) Minimize average time duration between trouble ticket creation to clear
- (Optimisation) Minimize diagnosis to complete
- (Optimisation) Minimize time spent in pending state
- (Optimisation) Minimize time spent on diagnosis

- (Goal) Maintain efficient complaint handling systems
- (Goal) Maintain high resolution speeds
- (Goal) Achieve high satisfaction amongst customers
- (Goal) Achieve high customer engagement
Knowledge Base artefacts from different levels of abstraction generally are described in different languages and with differing levels of coarseness. It is essential to maintain a knowledge base that describes key concepts and can be used to translate between the strategic and operational concepts. A domain specific knowledge base that describes the call centre is provided below as a set of rules are written as a knowledge base i.e. $A \Rightarrow B$.

- **Resolve(complaint) ∧ (Execution(time) ≤ 10min) ⇒ Resolution(complaint,high-speed)**

  This rule is saying that if a complaint is resolved in less than 10 minutes then it is considered to be high-speed resolution.

- **$\frac{|\text{instance(Resolution(complaint,high-speed))}|}{|\text{instance(all)}|} ≥ .55 \Rightarrow \text{Maintain-High-Resolution-speeds}$**

  This rule is saying that if the ration of instances that are completed at high-speed is greater than the threshold then the strategic goal is realized.

- **$\frac{|\text{instance(Resolution(complaint))}|}{|\text{instance(all)}|} ≥ .75 \Rightarrow \text{Efficient(ComplaintSystem)}$**

- **Efficient(ComplaintSystem) ⇒ Maintain-efficient-complaint-handling-systems**

- **Processed(case) ∧ Satisfied(customer) ⇒ Achieve-high-satisfaction-amongst-customers**

- **$\frac{|\text{instance(Engaged(customer))}|}{|\text{instance(all)}|} ≥ .75 \Rightarrow \text{Achieve-high-customer-engagement}$**

There are different types of languages that can be used to represent the information in the knowledge base. In this case we represent knowledge at an instance level, design level and strategic level. The languages used for each level are not based on any one standard. For ease of understanding: instance level knowledge is written in green, design level knowledge is written in blue and, finally strategy level knowledge is written as classic propositions and are marked in orange. Using this type of knowledge base with a framework such as that described in [148] gives organisation greater flexibility while conducting strategic analysis.

**Scenario**: PAC has conducted a detailed business analysis project to obtain all of the above BPM, Strategic and KB artefacts. It now seeks to leverage the information to become more adaptive a dynamic operating environment. In particular when there is large network outage in a particular area, the call centre experiences heavy complaints. When this happens, customers have high wait times and become disgruntled. This is evident through a high churn rate (a customer churns from a network by switching carriers). As a strategy, PAC would like to lower these churn
occurrences and has invested heavily in a scalable call centre (during peak demands, the call centre requests resources from external call centres or other call departments); however, the additional call centre team are not as flexible or knowledgeable about the complaints process as existing staff members.

3.4 Task Composition and Choreography

3.4.1 Semantic Effects and Accumulation

A semantic effect $e$ (or effect) is a partial state description for a given activity or event in a process. A set of effects $E$ denotes a set of all CNF sentences of a finite language $L$. We refer to a set of effects as an effect scenario. To disambiguate the language we often will call an effect scenario for an activity the immediate effect scenario of the activity, an effect scenario from the start of a process to any other point in a process model is called the cumulative effect scenario. Finally, an end effect scenario is an effect scenario found by computing state changes from the beginning of a process through all paths to a given end event.

For instance, given a process $p$ with the sequence of activities and events $\langle \psi, n_1, n_2, \varphi \rangle$. Where $\psi$ is a start event, $n_1$ and $n_2$ are activities, and $\varphi$ is an end event. To compute the effect scenarios up until activity $n_2$, we would accumulate an immediate effect scenario of $\psi$ with an immediate effect scenario of $n_1$; a result of which would then be accumulated with the immediate effect scenario $n_2$.

Each annotation can be accumulated using a function to produce a semantic description of the process model. Given two sets of effect scenarios $\epsilon_i$ and $\epsilon_j$, let a function $\text{acc}(\epsilon_i, \epsilon_j)$ return the accumulation of both sets of effect scenarios.

**Definition 3: Pair-wise accumulation**

Let $\text{acc} : 2^{L_e} \times 2^{L_e} \rightarrow 2^{L_e}$ be a function that takes as input two sets of effect scenarios and returns the resulting accumulation as a sets of effect scenarios where $L_e$ is the set of all well-formed sentences in the underlying language in which effect scenarios are described. The function $\text{acc}$ is defined over possibly infinite sets of effect scenarios $\epsilon_i$ and $\epsilon_j$ for all indices $x \in S$ and $y \in T$ of the sets of effect scenarios $\epsilon_i$ and $\epsilon_j$ where $KB$ denotes a domain knowledge base given in the language $L$.
3.4. Task Composition and Choreography

\[ acc(e_{ix}, e_{iy}) = \begin{cases} \\
\epsilon_{ix} \cup \epsilon_{iy} & \text{if } \epsilon_{ix} \cup \epsilon_{iy} \cup KB \nvdash \bot, \text{ otherwise} \\
\epsilon'_{ix} \cup \epsilon_{iy} & \text{where } \epsilon'_{ix} \subseteq \epsilon_{ix} \text{ s.t. } \epsilon'_{ix} \cup \epsilon_{iy} \cup KB \nvdash \bot \text{ and } \\
& \text{there exists no } \epsilon''_{ix} \subseteq \epsilon'_{ix} \subseteq \epsilon_{ix} \text{ where } \epsilon''_{ix} \cup \epsilon_{iy} \cup KB \nvdash \bot 
\end{cases} \]

As an example of the machinery for computing effect scenarios, consider the following sequence of activities \( \langle a_1, a_2, a_3 \rangle \). Where the immediate effect scenario for completing activity \( a_1 \) is \( E_{a_1} = \{ \alpha \} \). The immediate effect scenario for completing \( a_2 \) is \( E_{a_2} = \{ \beta \} \), and the immediate effect scenario for completing \( a_3 \) is \( E_{a_3} = \{ \gamma \} \). The cumulative effect scenario for activity \( a_3 \) can be computed using \( acc(acc(\{ \{ \alpha \} \}, \{ \{ \beta \} \}), \{ \{ \gamma \} \}) \). First, the cumulative effect scenario of activities \( a_1 \) and \( a_2 \) are determined; then, using this cumulative effect, we accumulate it with the immediate effect scenario of \( a_3 \).

As a second example: given the same three activities and a knowledge base \( KB = \{ \gamma \rightarrow \neg(\alpha \land \beta) \} \) where the immediate effect scenario at \( a_1 \) is \( \{ \alpha, \beta \} \). The immediate effect scenario at \( a_2 \) is \( \{ \gamma \} \) and the immediate effect scenario at \( a_3 \) is \( \{ \alpha \} \) then the cumulative effect scenario at \( a_2 \) is computed by \( acc(\{ \{ \alpha \land \beta \} \}, \{ \{ \gamma \} \}) \). The cumulative effect scenario at \( a_2 \) is \( \{ \{ \alpha, \gamma \}, \{ \beta, \gamma \} \} \). To compute the end effect scenarios for the process at \( a_3 \), each of the possible effect scenario in the set of cumulative effect scenarios are substituted into the \( acc \) function i.e. \( acc(\{ \alpha, \gamma \}, \alpha) = \{ \alpha, \gamma \} \) and \( acc(\{ \beta, \gamma \}, \alpha) = \{ \{ \alpha, \gamma \}, \{ \alpha, \beta \} \} \). Therefore the set of end effect scenarios resulting after this accumulation is \( \{ \{ \alpha, \gamma \}, \{ \alpha, \beta \} \} \).

3.4.2 Semantical Annotated Processes

There are various encodings of process models. Typically, the encodings either find a basis in the space of Petri nets or graphs. The most common encoding for process models is a WF-net (workflow network) described by van der Aalst [197]. These are not suitable in this context as they describe execution flow from only one start point. Additionally they are too general to describe concurrent flow across multiple instances. As an alternative, Dijkman et. al. [50] have described a graph encoding of the BPMN 2.0 standard that takes into consideration the execution semantic of a large selection of BPMN elements. In this thesis, a generalisation of Dijkman’s encoding is proposed.

A process model as a graph is a strongly connected directed graph, where activ-
3.4. Task Composition and Choreography

ities events and gateways are nodes in the graph and sequence flow are the edges. Each node in the graph is labelled with a type, a name, and extra attributes.

**Definition 4: Semantically Annotated Process Model**

A process model is a labeled directed graph \( p = (N, F, l, \Omega, \psi, \phi) \) with the following properties:

1. \( N \) is a finite non-empty set of nodes. \( \psi, \phi \) are the start and end nodes respectively and \( \psi, \phi \in N \).

2. \( F \) a set of control flow links, \( F \subseteq N \times N \).

3. \( \Omega \) is a set of labels, each label \( \omega \) is of the form \( \langle \text{type}, \text{name}, \langle v, e \rangle \rangle \). Where \text{type} is the type of element, i.e. event, activity, gateway, task. \text{name} is the name of the element. \( \langle v, e \rangle \) is a tuple representing the QoS capabilities and effect scenario of the node respectively.

4. \( l : N \rightarrow \Omega \) is a labeling function that assigns labels to nodes of the process model.

5. \( \forall n \in N, (n, \psi) \notin F \land (\phi, n) \notin F \) i.e., the start node has no incoming edges and the end node has no outgoing edges.

From our example, in Figure 3.2, each activity, event, pool, and gateway will be encoded as a node in a graph. Sequence and message arrows will be encoded as edges connecting two nodes, and annotations such as `efficient(system)` will be encoded as the effect scenario \( e \) for the labeled node `compare details of customer and complaint to business rules`.

A process model \( p = (N, F, l, \Omega, \psi, \phi) \) is well-formed if a strongly connected graph can be formed from its nodes and edges, i.e., \((N, F \cup (\phi, \psi))\) is a strongly connected graph. A well-formed process model \( p \) is a well-formed decision free task sequence if there are no XOR gateways in the process model. There are procedures that can be used to construct a collection of well-formed decision free task sequences from a given well-formed process model\(^1\). There is a description of such a procedure in §7.1.1. A set of well-formed decision free task sequences is referred to as a process portfolio \( \mathcal{P} \).

An example of a decision free version of the complaint handling process is \( \langle \{\text{Receive complaint}, \text{verify customer details}, \text{customer verified?}, \text{request customer attend local store}, \text{end event}\} \rangle \), \( \langle \{\text{Receive complaint}, \text{verify customer details}\} \rangle \), \( \langle \text{verify...} \rangle \).

\(^1\)A supporting toolkit of libraries implementing most functions described here can be downloaded from [http://www.dsl.uow.edu.au/~edm92/textseer/\(^2\)] including a procedure to create decision free task sequences, further §7 includes a discussion on the implementation of such an procedure.
customer details, customer verified?⟩, ⟨[customer verified?, request customer attend local store], [end event]⟩, Ω, [Receive complaint], [end event]), where Ω is the set of labels and associated effects, i.e. {⟨ Activity, [Verify customer details], ⟨∅, unhappy(customer)⟩⟩, ⟨ Activity, [Request customer attend local store], ⟨∅, ¬engaged(customer) ∧ ¬resolution(customer)⟩⟩}. Unless otherwise specified, all process models discussed beyond this point are assumed to be decision free task sequences. For the remainder of this chapter and this thesis we interchangeably use decision free task sequence, semantically annotated process model and process model to mean any model that can be encoded in the graph structure shown above.

3.4.3 Semantic Process Composition

The result of accumulating effect scenarios and QoS capabilities in process models is a set of effect scenarios and QoS capabilities that describe the entire process model. There are occasions when it is beneficial to find the effect scenarios or QoS capabilities of a particular instance (or trace) of a process model. A trace is a sequence of activities showing a possible execution instance of the given process. Each trace begins at the start of the process model and continues along to activities in the process until a given point within the process model. To find these traces, either sequential paths, parallel paths, or a combination of the two must be used to describe the instance of the process model of interest.

Given a decision free task sequence \( p = \langle N,F,I,\Omega,\psi,\varphi \rangle \), a path through the sequence is:
\[
\langle (n_1,n_2), (n_2,n_3), \ldots, (n_{j-1},n_j) \rangle
\]
where elements of the path are control flow links and each \( n_x \) in the path is distinct. We shall say \( n_i \prec n_j \) iff there exists a path \( \langle (n_i,n_{i+1}), \ldots, (n_{j-1},n_j) \rangle \) where \( n_i \) precedes \( n_j \).

Definition 5: Neighbor Function

Let \( \text{Neighbor}_p(n_i) : N \rightarrow 2^N \) be a function that returns the neighbor of any node \( n_i \) in a decision free task sequence \( p \).

\[
\text{Neighbor}_p(n_i) = \{ n_j \mid n_j \neq n_i \land \left( ((n_i,n_j) \in F) \lor (n_i \not\prec n_j \land n_j \not\prec n_i) \right) \}
\]

Definition 6: Trace

Given a decision free task sequence \( p = \langle N,F,I,\Omega,\psi,\varphi \rangle \), a trace is a sequence \( \sigma = \langle n_1,n_2,\ldots,n_m \rangle \) where:
3.4. Task Composition and Choreography

- \(|N| = |\sigma|\).
- For each pair \(\langle n_i, n_{i+1} \rangle\) in \(\sigma\), \(n_{i+1} \in \text{Neighbor}_p(n_i)\) for \(1 \leq i \leq m\).
- For any two nodes \(n_i, n_j \in \sigma\), where \(i < j, n_j \not\prec n_i\).

The set of all traces of a decision free task sequence is \(\Sigma_p\).

For any two nodes \(n_i\) and \(n_j\) in a trace \(\sigma\), if \(\sigma = \langle\ldots, n_i, \ldots, n_j, \ldots\rangle\) we say \(n_i\) comes before \(n_j\). If for all traces \(\sigma \in \Sigma_p\) of a given process model \(n_i\) comes before \(n_j\) then we say that \(n_i\) always comes before \(n_j\) and denote this \(n_i \ll \Sigma n_j\).

Cumulative effects and QoS values are computable over decision free task sequence traces. Accumulation of effects and QoS values have been discussed in previous work [95, 98, 115, 148] and previously in §3.4.1, the result of which is a consistent set of effects and QoS values that give a semantic meaning to a process model. An accumulation function \(\text{accumulate}\) takes as input a trace \(\sigma\) and returns a tuple \(\langle \Upsilon, \epsilon, \Gamma \rangle\) which are the set of cumulative QoS values, the set of cumulative effects and the cumulative customer state respectively. This is achieved by pairwise accumulation of tasks across the trace.

**Definition 7: Case Sequence**

A case sequence \(C\) is a tuple:

\[
\langle \mathcal{M}, \Pi, \mathcal{A}, N, F, I, a, \Omega, \psi, \varphi \rangle
\]

Where:

1. \(\mathcal{M}\) is a set of decision free task sequences that are part of the case sequence.
2. \(\Pi\) is a sequence of traces \(\Sigma_p \in \mathcal{M}\).
3. \(\mathcal{A}\) is a set of tuples \(\langle \Upsilon, \epsilon, \Gamma \rangle\) of QoS values, accumulated effects, and customer environment.
4. \(N\) is a finite non-empty set of nodes. \(\psi, \varphi\) are the start and end nodes respectively and \(\psi, \varphi \in N\).
5. \(F\) a set of control flow links, \(F \subseteq N \times N\).
6. \(\Omega\) is a set of labels, each label \(\omega\) is of the form \(\langle \text{type}, \text{name}, \langle \upsilon, \epsilon, \gamma \rangle \rangle\). Where type is the type of element. name is the name of the element. \(\langle \upsilon, \epsilon, \gamma \rangle\) is a tuple representing the QoS capabilities for the element, effect scenario of the element and customer state respectively.
7. \( a : \Pi \rightarrow A \) is a function that maps trace sequences to accumulation tuples.

8. \( l : N \rightarrow \Omega \) is a labelling function that assigns labels to nodes of the case sequence.

9. \( \forall n \in N, (n, \psi) \notin F \land (\varphi, n) \notin F \) i.e., the start node has no incoming edges and the end node has no outgoing edges

For any case sequence \( C \), if all \( p \in M \) are in a process portfolio \( P \) then we say that \( C \) belongs to \( P \), denoted \( C \sqsubseteq P \). To compute the accumulated effects and QoS capabilities for a case sequence, each sequential pair of traces \( \langle \sigma_i, \sigma_j \rangle \in \Pi \), where \( \langle \upsilon_j, \epsilon_j, \gamma_j \rangle \) are the QoS, effect values, and customer states for the start node of \( \sigma_j \) first pairwise-accumulate \( \langle \upsilon_i, \epsilon_i, \gamma_i \rangle \) and \( \langle \upsilon_j, \epsilon_j, \gamma_j \rangle \) then accumulate across the remains of the trace \( \sigma_j \). §3.4.1 describes pairwise-accumulating cumulative effects with effects. The result of this process is a cumulative effect \( \langle \upsilon_P, \epsilon_P, \gamma_P \rangle \).

Returning to our example, §3.3, where PAC wishes to recompute their operational stack to meet the demands of their changing environment. In the first instance, it would be ideal to be able to recompute the process model that they follow, to better manage their workforce. To recompute a process model, it’s graph encoding needs to be mathematically analysed. If we wanted to determine the case wide effect for a customer who had phoned PAC to make a complaint but due to staffing issues was unable to be verified. We would first create a decision free task sequence of the path that the customer had taken through the process. Then compose the process with any other processes that the customer may have followed as part of their engagement with PAC. A cumulative semantic effect of \( \{ \text{unhappy(customer)}, \neg \text{engaged(customer)}, \neg \text{resolution(customer)} \} \) can be found. By computing QoS based artefacts such as engagement time we can compute the QoS effect, e.g. \( \{10\text{min}, \$-10 \text{staffing}\} \). Finally a case based effect could also be carried through \( \{80\% \text{churnRisk}, \$-50 \text{goodwill}\} \).

When describing an adaptive case management system an a difficulty exists of describing cases and caseflow across differing levels of abstraction (between perennial non-operational soft workflows and transient processes). In this framework, we propose that composing processes to correct the degree of abstraction is the best way to address this issue. Composition has been choosen here because the decomposition requires a pool of potential predefined sub-processes or atomic tasks (or requires extensive insightful thoughts by an analyst). Further to this, in the event of an organisational change, or in a change of understanding, transforming the purpose of multiple layers of processes and their decomposition becomes cumbersome and ex-
pensive. An unrefined process is compound, and due to this that it is often necessary to consider the sequencing of processes or parallel execution of processes. When fused together, these composed processes may create a complex enough expression to consider the expression a case.

Given any number of process models with the purpose of investigating the consequences of running them in sequence or parallel, it becomes necessary to understand both the QoS value and semantic effect of the composed processes to gain insights. We compute accumulation over the composed process by manipulating processes and case sequences using a sequential combination operator and a parallel combination operator which both translate the composed processes into case sequences which we have already shown an accumulation procedure for. We will denote pairwise case sequence composition accumulation as \( C_i \sqcup C_j \). Sequential process composition, requires the selection of a trace from the set of possible traces for both of the processes. The end event of the first process is then converted to an intermediate event and joined to the start (converted to an intermediate event) of the selected trace from the next process in a series. Each new sequence of nodes shows an end-to-end arrangement of a composed process. Using pairwise accumulation along each sequence it is possible to compute the of effects and QoS values for the composed process. The joining of two processes in sequence is denoted by the operator \( \otimes \).

To find parallel traces, we essentially join two processes into a parallel design. Converting the start and end events from each into intermediate events and placing the processes between parallel gateways. Using the methods in definition 6 to compute the set of traces for the composed process it is easy to find the end-to-end arrangement of the composed process. The joining of two processes in parallel is denoted by the operator \( \oplus \).

Due to the nature of the sequential composition, in a case sequence composition description \( \oplus \) has precedence over \( \otimes \), i.e. \( p_1 \otimes p_2 \oplus p_3 = p_1 \otimes (p_2 \oplus p_3) \neq (p_1 \otimes p_2) \oplus p_3 \). The composed models generated through these procedures is a case sequence.

Given a process portfolio \( \mathcal{P} \), we assume there exists a number of case sequences \( C_1, C_2, \ldots \) that can be constructed by composing various processes and case sequences together. An accumulation function \( \text{accumulate} \) provides a method for semantic definition from each \( C_i \) to some tuple \( a = \langle \Upsilon, \mathcal{E}, \Gamma \rangle \), i.e. \( \text{accumulate}(C) = a \) where \( a \) is a closed wff, and due to the nature of effect accumulation \( \text{Th}(\text{accumulate}(C)) = \text{accumulate}(C) \).

Let a base case sequence \( B_\mathcal{P} \) for a particular process portfolio \( \mathcal{P} \), be any case sequence \( C \sqsubseteq \mathcal{P} \) with \( U = \{ p \mid p \in \mathcal{M}_C \} \) where there does not exist another case sequence \( C' \sqsubseteq \mathcal{P} \) with \( V = \{ p \mid p \in \mathcal{M}_{C'} \} \) where \( V \subset U \).
3.5 Conclusion

Given a set of base case sequences \( \{B, B', \ldots \in B_P\} \), an extension case sequence of a base case sequence \( B \) with \( W = \{p|p \in M_B\} \), is any case sequence \( C \subseteq P \) and \( C \not\in B_C \) with \( U = \{p|p \in M_C\} \) where \( W \subseteq U \) and there does not exist another case sequence \( C' \subseteq P \) and \( C' \not\in B_P \) with \( V = \{p|p \in M_{C'}\} \) where \( V \subseteq U \) and \( W \subseteq V \) and there exists an operator \( \oplus \) or \( \otimes \) where \( B \oplus B' = C \) or \( B \otimes B' = C \). A child extension is some case sequence \( C^* \) that can be found by forming a chain of extension case sequences, i.e., if \( C \) is a base case sequence and \( C' \) is an extension case sequence for \( C \), and \( C'' \) is an extension case sequence for \( C' \) etc., until \( C^* \).

**Definition 8: Case Sequence Accumulation**

Let \( q = \langle \Upsilon, \epsilon, \Gamma \rangle \) be a tuple describing some set of QoS and effect scenarios and \( C \) be a case sequence. Let some base case sequence \( C_0 \) have an associated \( q' \) where \( q' \not\|= q \) and for each \( i \geq 0 \) where \( C_{i+1} \) is an extension case sequence of \( C_i \):

\[
\text{accumulate}(C_{i+1}) = \text{Th}(\text{accumulate}(C_i)) \cup C_{i+1}
\]

Then \( q \) is the semantic description and cumulative effect of \( C \) iff: \( q = \psi^\infty_{i=0}(C_i) \) and \( C \) is a child extension of \( C_0 \).

By finding case sequences, and then using an accumulation function, it is easy to describe the effect scenarios or QoS capabilities of a multiple processes instance, i.e., if multiple processes were used to handle a particular case, then a QoS capabilities like time taken can be computed. It is also possible to use semantic effect annotations to provide contextual information on the state of a case at any given point during the processing of the case across any number of processes.

### 3.5 Conclusion

In this chapter, we have provided a method to compute case sequences from a collection of process models. Elements on the system described have been developed into a prototype library\(^2\). The result and benefit of using a case management system formed from existing legacy process management systems are that transition and change costs will be dramatically reduced for the organisation. The results of moving towards adaptive case management using our framework will provide organisational case managers an apparatus to understand the current case state of affairs across the entire operational context. The framework that we have presented contributes to a better understanding of adaptive case management, and further tool support will

\(^2\)The source code for the framework can be found online at [http://www.dsl.uow.edu.au/~edm92/](http://www.dsl.uow.edu.au/~edm92/)
3.5. Conclusion

equip decision makers with a device to understand the sustainability of this technology in an operational context.
Business Process Integration

If you want to inspire confidence, give plenty of statistics it does not matter that they should be accurate, or even intelligible, so long as there is enough of them...

CHARLES DODGSON

Process integration is an activity where analysts investigate relationships across process repositories to classify and merge similar activities into a standardized system. Integration is a problem that affects analysts working on existing legacy systems, where there is a need to consolidate. The steps in process integration include identifying relationships in the system, merging related activities and then producing a general consolidated process. Process matching is the most understood and researched area in the integration space. Generally matching can be defined as “process model matching refers to the creation of correspondences between the activities of the process model” [33]. Or, formally “Business process matching is the procedure of finding a partial injective mapping $M$ of nodes $n_i \in P_i$ to nodes $n_j \in P_j$ such that some real functional score is maximal” [176].

Similarity between processes can be simply defined as “two process models $P_i$ and $P_j$ [are] similar, if they expose a common share of behavior” [121], simply there are activities common to both process models. Automated systems for finding related process are important and can weigh on business analyst activities. In [176], Rosa et. al. demonstrate the cost of manual merging showing that manually merging similar processes can be both costly and time-consuming. This chapter describes a practical method for process similarity measurement. Also in this chapter, I provide a theoretical framework and measures to help the process of integration.
4.1 Need for Process Integration

Business process integration is a problem in a wide variety of domains. Namely in a commercial setting where business and systems analysts are commonly required to model system consolidations. Consider for example a financial service administrator. The administrator must maintain member management processes for their clients. The processes include member onboarding, member details change, account roll-over to and from other funds, and account consolidation. The administrator must support different versions of these processes for each client through the closely the processes are integrated, the lower the operational cost for the administrator i.e. benefiting from economies of scale. It is typical for slight variances in each fund process. Variances occur because each fund has different goals that they aim to achieve with their process. The administrator, however, seeks to rationalize variants to a general process as this provides them with a competitive advantage. Providing general processes to client funds at a lower price is an effective way to leverage the economies of scale. The difficulty in this activity is rationalizing a process that can achieve the goals of all client funds. Consider another example, where an insurance company buys a smaller insurance company. The resulting entity must support a single claim handling process. An integrated claims process for both companies will need to meet the goals of both companies. Over time, the claims process is likely to be integrated into a single claims process as the smaller company is completely absorbed.

In both examples, the consolidated process must be as similar as possible to the original processes. Focusing on similarity workers suffer less disruption, and the change process is less risky.

We defined the business process integration problem as the problem of identifying a single process that:

1. Achieves all of the goals/objectives of a set of prior processes while
2. Minimizing the extent of change required to the original processes.

We have created a general process integration methodology that satisfies these two properties. The first part of methodology involves the identification of goals that the integrated process must achieve. There are many methods for this, the simplest of which is taking the intersection of the goals of the antecedent processes that must be integrated. We will address goal integration in section §4.2. The second part of the methodology is to measure the extent of change between a process variant and the original process. The aim of which is to determine the most minimally different
potential successor. In part §4.4, we will present an approach to business process integration based on proximity measures. An assumption of this chapter is that processes have BPMN syntax and include semantic annotation.

Lightweight annotations are beneficial to process analysts to document effects during the design phase. Bearing in mind that heavier formal methods for annotation would find low acceptance in practice. Also, translation of models into formal semantic domains would be impractical on most projects. Based on this assumption, a uniform graph-based encoding of annotated BPMN models is used. Then a class of process proximity measures is described showing how these can form the basis for effective business process integration.

4.2 Approaches to Integration

The principal purpose of any model is to “identify the structural features that have the greatest implications for policy, and thus are worthy of further pursuit” [60]. We use business process modeling as a means to express the operation of organisational systems, based on a combination of artifacts and knowledge extracted from domain experts. Maintenance of the formal system can be viewed as a problem to be solved within the notation.

Matching

Process matching is the process of clustering and relating similar activities. These clusters can be derived using various methods each with strengths and weaknesses that can leverage the knowledge stored in a process.

Clustering techniques classify objects (such as business process models) into partitions so that the data in each subset share common traits. [102] outlines a number of clustering methods and functions such as the k-mean algorithm. During the clustering phase, each element is classified into a group of related elements. In cases where data cannot be classified easily using large dataset averaging methods, the classification of objects in a particular domain can be completed by separating objects into classes based on their attributes, and giving criteria for determining whether a particular object in the domain is in a particular class or not. An example of this is bi-clustering [27]. Bi-clustering is a clustering approach where a set of samples are simultaneously partitioned into subsets in a way that they have a high relevance to each other. The problem of using these techniques within an organisational domain is the complexity associated with implementations. Most implementations of data
4.2. Approaches to Integration

clustering are implemented in large-scale projects such as gene mapping and search engine crawling.

Smaller steps can be taken to reduce the complexity of large-scale data classification requirements with the use of naming conventions. For example, activity names should be at most two or three words long and carry clear and concise meanings providing an unambiguous label that all domain specialist would understand. Each data set name will provide a significant meaning to the observer. During design, analysts define models using meaningful naming conventions to provide clarity in some context. Kementsietsidis investigates methods for the logical design of data values to promote integration from heterogeneous data sources using data mapping tables. The tables maintain correspondences between, for example, business processes within a process repository. Thus, queries may result in alternate names, retaining knowledge in a particular domain.

SISIBIS is an example of classification completed by system users in the collaborative database schema integration tool. During the creation of enterprise data schemas, the platform required that analysts and system users tag elements. Users were instructed to allocate tags with semantic meanings and describe how various data was designed using contextual descriptions.

The use of matching techniques to connect elements from processes helps reveal contextual similarity. Contextual similarity is required in process integration, acting as a mapping function that shows direct similarities between activities in processes. In Dongen, et al. presents a vector based proximity metric for contextual similarity. These metrics show the operational context of a sequence of activities. They are found by clustering activities based on semantic similarities across documents (using causal and contextual footprints and combining with semantic scoring) over a business domain vector of semantic artifacts. This approach allows an analyst to rank semantic equivalences between two or more processes. The adjunct system described in (Mendling) shows structural integration methods using Event-driven Process Chains (EPC’s) against some SAP process models. In this work a structure merge operator is defined for use on SAP models that can be used once a semantic similarity between functions has been defined. Mendling also shows a reduction method that can be achieved by eliminating redundant process pathways while keeping EPC based structural integrity.

In the preceding research, EPCs are used to verify structural ‘soundness’ or non-recoverable errors as well as reducing complexity within the structure. The problems associated with relying on a union combination domain artefacts is in con-
4.2. Approaches to Integration

Considering the similarity of functions defined using synonyms and words, or similarly of footprints and ignoring the frequency of the metrics \[130,200\]. We address these problems by considering differences and adding together algebraic distances. In \[141\] Mendling notes “EPCs offer OR-joins which cannot be mapped to Petri nets without losing readability, there is a more general approach needed”. This is a limitation of all integration methods incorporating EPCs with behaviour integration schema’s as they can not be mapped without data loss. For integrating datasets that fall outside of each other context or for data set that overlap context various names may convey meanings that have not been meant by the analyst. There exist some solutions to this problem. In \[110\] Kementsietsidis investigates methods for logical design of data values to promote integration from heterogeneous data sources using data mapping tables. This is done by focusing on data values and how values correspond. The aim being to “to correctly restructure and map data” by “sharing their schemas and cooperating in establishing and managing the queries”. Here as a query is run on a data element, results with alternate names will also be shown. This method for naming conflict resolution will retain knowledge for a particular domain, however it may lead to problems.

An alternate method for data dissemination is in \[155\] whereby an organisation adopts a standardized data-entity naming convention. Here the language to be finally adopted must satisfy micro and macro naming requirements. A micro naming strategy defines how words and letters are syntactically arranged for a name (i.e. Syntactically the choice of hungarian notation for data type definitions iVar would mean a variable of type integer). Macro naming strategies refers to the relationsips of names to other data structures and logical data (i.e. ID, may have reference to Customer Identification Number in a sales lead process whereas; in an employer services process the name ID may refer to an company). The objective of a naming scheme must satisfy the user requirement. If the user has a question in mind then using a scheme the system should be able to retrieve information so that these question, as well as any new questions that come to light after considering the new information presented, are answered. In \[129\] Lewis suggests that retrieval should be considered a natural language problem rather than an excercise in controlled language search. This of course makes process integration a much tricker task. For our proposal we assume that incoming processes are described using a controled language.

Integration

Integration is the process of merging elements of two similar antecedent processes to create a single process that can be used to replace the original processes. Integra-
4.2. Approaches to Integration

Integration can be generalised into two activities, aggregation and regression. Aggregation is the process of combining data elements after detecting common elements or common relations [209]. This is done in its simplest form by combining common elements from two antecedent processes.

Process integration aims to investigate relationships across a business compendium to produce classifications and merge activities into a standardized system. This involves both matching and merging methods. The process of integrating various activities is largely based on matching criteria. Once objects are considered close enough to integrate with one another, and if each object is not equal to the other, then the merging process will begin.

Hinke [96] shows a further depth to aggregation by comparing general cardinality aggregations and inference aggregation in which predictions of inference emerge from data analysis activities. Here, not only are similar activities from antecedent processes joined in an integrated output. There is also a case where if an antecedent activity has a relation to an activity that does not have a direct role in a process but acts as a constraint on future activity within the process, then the activity is included during integration as an inference activity. For example, consider a process where there is an activity of ‘stamping letters in a mailroom’. During the integration of two processes that describe ‘sending a letter to a customer’, we must consider ‘stamping a letter’ as a constraint to be satisfied before ‘mailing the letter’. This activity should be included in the integrated output process even if it is not explicitly defined in one of the antecedent processes.

Regression is a stage within an integration system that involves reduction of the possible resulting process solution space while maintaining consistency. Regression in the use of process integration is useful for selecting optimal solutions. As a model of a process is aggregated possible solutions emerge. It is during regression that duplicate and structurally unnecessary data and information is removed to form explicit processes. These processes then become potential candidate implementation process.

In [145] Morimune offers some regression testing methods that can be used to for the creation of these candidate processes, using homogenous constraints.

In [87] Grossmann characterises integration operators by classify the properties of process operators and their resulting integration possibilities. Grossmann examines the possibilities that result from merge operators that be used to generate combinations of a solution space. Continuing in [88] and [86], Grossmann proposes a meta framework to support behaviour based integration. Allowing an analyst to validate and compare their merge solutions with potential models to find short comings or de-
tect potential flaws within a solution. Again this work is based on providing characteristics of potential results as a basis for a activity selection criteria based on semantic relationship and integration options(patterns).

Research into the area of business process space has resulted in interesting work where many of the technical aspects of integration have been addressed. In [87], a method for business process integration is presented which relies on the introduction of detailed and explicit process states, inter-process dependencies, and synchronizations as integration criteria. In comparison, the work in this chapter presents a goal and proximity-directed criterion (relying on minimal analyst intervention) allowing analysts to explore candidate integrations that maintain structural and semantic similarity to their antecedents. In [141], a (database) view integration-inspired business process integration method achieved via a view-merge operator, identity/ordering relations, and restructuring (or simplification rules) is presented. In comparison, we outline criteria that help establish identity relations and minimize structural and (some) semantic differences during integration.

We have addressed technical challenges of integration though it is important to mention the social side of integration. The characteristics of an integrated process will be impacted by the goals of an organisation as well as the goals of stakeholders. Kulik [120], makes the case that organisations are functionally goal orientated, systematic, and structurally complex. During business process integration an organisation should consider the socioeconomic implications of certain groupings and their effects across an organisation [101]. This part of the integration process is important to note and consider during implementation of an integration project as the effects of integration on the human elements within a system are the most susceptible to failure.

Generally we have found that keeping hierarchal levels (views) of processes is best considering each organisation as a society. If we view an organisation as a simulation environment as in [120] and consider the ideas of belief merging proposed in [39] where the correlations of social theory and computational belief merging are examined. Here the individual preferences are aggregated to obtain societal ones. There is also an examination of strategy proof merging where the goal is to define procedures that most accurately and fairly represents the preferences of the population.

It is also interesting to note the effect of the process cluster size. Where if we consider all processes as a generalized system and then preform integration operations on them we may discover certain traits that are applicable, conversely if we consider strict and narrow process chains we limit our ability to find organisational optimiza-
In the following sections we provide a conceptual framework that can be relatively easily implemented in decision-support tools to determine degree of similarity of process model integration options. A key challenge with BPMN is that it provides relatively little by way semantics of the processes being modeled. Another challenge is that there is no consensus on how the semantics of BPMN might be defined, although several competing formalisms have been proposed. Since integration clearly requires more information than is available in a pure BPMN process model, we propose a lightweight, analyst-mediated approach to semantic annotation of BPMN models, in particular, the annotation of activities with effects. Model checking is an alternative approach, but it requires mapping BPMN process models to state models, which is problematic and ill-defined. We define a class of proximity relations that permit us to compare alternative modifications of process models in terms of how much they deviate from the original process model.

![Image of Customer Verification Process 1]

Figure 4.1: Customer Verification Process 1

### 4.3 Annotated Process Nets

Semantic Process Nets (SPNets) were originally presented in [71] and provide us with a uniform structural and semantic encoding of a BPMN model. Using SPNets we began developing theory for business process integration, this was previously discussed in [147].
4.3. Annotated Process Nets

Definition 9: (SPNet)

A Semantic Process Network (SPNet) is a graph \( \langle V, E, s, t, l_V, l_E \rangle \) such that: \( V \) is a set of nodes; \( E \) a set of edges; \( s, t : E \rightarrow V \) are source and target node mappings; \( l_V : V \rightarrow \Omega_V \) maps nodes to node labels; and, \( l_E : V \rightarrow \Omega_E \) maps edges to edge labels. Each label in \( \Omega_V \) and \( \Omega_E \) is of the form \( \langle \text{id}, \text{type}, \text{value} \rangle \).

We note that a unique SPNet exists for each model in BPMN. This can be determined objectively through transformation. Each event, activity or gateway in a BPMN model maps to a node, with the \text{type} element of the label indicating whether the node was obtained from an event, activity or gateway in the BPMN model.

The \text{value} elements for immediate effect, and cumulative effect edges are triples of the form \( \langle \text{id}, \text{function}, \text{quality} \rangle \). The \text{id} element of an immediate effect edge corresponds to the source node \text{id} label element. The \text{id} element of a cumulative effect edge is a scenario identifier (a vector) where each element is either: a node identifier; or, a set whose elements are (recursively) scenario identifiers. A scenario identifier describes the precise path that would have to be taken through the process model to achieve the cumulative effect in question.

The \text{function} element of an immediate effect, or cumulative effect edge label is a set of assertions, whereas the \text{quality} element is a vector of QoS evaluations. The \text{function} and \text{quality} elements of an immediate effect annotation edge label can be viewed as a context-independent specification of its functional and non-functional effects. These must be accumulated over an entire process to be able to specify, at the end of each activity, the contextual \text{function} and \text{quality} elements of cumulative effect annotation labels. These labels indicate the functional and non-functional effects that a process would have achieved had it executed up to that point.

Case Sequences, defined in Definition 7 in §3.4.3 is an evolution of the original SP-Net, composed of nodes, edges, and annotations. In particular the more general case
sequence contains further information on QoS effects and composite process definitions and has been defined for effect accumulation across multiple processes.

### 4.4 Semantic Process Net Integration

Business process proximity is used during integration to establish a distance measure between two or more case sequences. Intuitively, this measure is used to ensure that the integrated model is as similar as possible to its antecedents. In other words, we would like to minimize the deviation of an integrated model from its ancestors, thereby utilizing the previous legacy configuration and minimizing effort during integration.

Traditionally, processes similarity has been shown using graph edit distance over two process models $G$ and $H$. Business process matching is the procedure of finding a partial injective mapping $M$ of nodes $n_i \in p_i$ to nodes $n_j \in p_j$ such that some real functional score is maximal [125,176]. Alternatively [51] “assume two process models $p_i$ and $p_j$ to be similar, if they expose a common share of behavior”.

In order to compute our structural distance measures, we consider the set of nodes and the set of edges of the models in the following way:

**Figure 4.3: Example Integrations**
4.4. Semantic Process Net Integration

- \( d_N(N_i, N_j) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|} \); (Jaccard coefficient)

- \( d_F(F_i, F_j) = \frac{|F_i \cap F_j|}{|F_i \cup F_j|} \); (Jaccard coefficient)

- \( D_N(N_i, N_j) = |N_i| + |N_j| \)

- \( D_F(F_i, F_j) = |F_i| + |F_j| \)

Associated with each pair of processes is a proximity measure: \( d(p_i, p_j) \). When given an integrated process \( p_i \), and one of its antecedents \( p_j \), computes the distance of \( p_i \) from \( p_j \) w.r.t. a structural criteria alternatively defined as either (or by combining):

- \( d_N(N_i, N_j) + d_F(F_i, F_j); \)

- \( w_N d_N(N_i, N_j) + w_F d_F(F_i, F_j); \)

- \( \frac{d_N(N_i, N_j)}{D_N(N_i, N_j)} + \frac{d_F(F_i, F_j)}{D_F(F_i, F_j)}; \)

such that: \( d_N \) and \( d_F \) are nodes and edge proximity measures; \( w_N \) and \( w_F \) are weights for each metric; and, \( D_N \) and \( D_F \) indicate the maximum hypothetical distance between nodes and edges.

We can then extend, and incorporate the process proximity measures into a case sequence proximity measure that can be used to measure the distance of case sequences both structurally and semantically.

In order to compute our structural distance measures, we consider the set of traces in the following way:

- \( d_\Pi(\Pi_i, \Pi_j) = \text{damLev}(\Pi_i, \Pi_j) \); Where \( \text{damLev} \) is the Damerau–Levenshtein distance.

- \( D_\Pi(\Pi_i, \Pi_j) = |\Pi_i| + |\Pi_j| \)

Computing semantic proximity \( d_{acc} \) is somewhat more complicated as it relies on the possible end effect (outcome or scenario) of both case sequences. Recall that an end effect scenario of a case sequence can be computed with the function \( \text{accumulate}(C) \), which results in sets of effect scenarios. For each set of effect scenarios \( e \) in \( \text{accumulate}(C) \), we create a term frequency-inverse document frequency vector of effects in each effect scenario \( \bar{N}_C \) containing \( tf - idf(e, \cup E) \) for all \( E \in \cup e \in \text{accumulate}(C_i) \cup \text{accumulate}(C_j) \).

Associated with each pair of case sequences is a proximity measure:

\( d(C_i, C_j) \)
4.4. Semantic Process Net Integration

Which given an integrated case sequence \( C_i \) and one of its antecedents \( C_j \), computes the distance of \( C_i \) from \( C_j \) w.r.t. a combination of structural and semantic criteria alternatively defined as either (or by combining):

\[
\begin{align*}
& \cdot d_N(N_i, N_j) + d_F(F_i, F_j) + d_{\Pi}(\Pi_i, \Pi_j) + d_{acc}(C_i, C_j); \\
& \cdot w_N d_N(N_i, N_j) + w_F d_F(F_i, F_j) + w_{\Pi} d_{\Pi}(\Pi_i, \Pi_j) + w_{acc} d_C(C_i, C_j); \\
& \cdot \frac{d_N(N_i, N_j)}{D_N(N_i, N_j)} + \frac{d_F(F_i, F_j)}{D_F(F_i, F_j)} + \frac{d_{\Pi}(\Pi_i, \Pi_j)}{D_{\Pi}(\Pi_i, \Pi_j)} + \frac{d_{acc}(C_i, C_j)}{D_{acc}(C_i, C_j)},
\end{align*}
\]

such that: \( d_N, d_F, d_{\Pi}, \) and \( d_{acc} \) are node, edge, trace and semantic (effect) proximity measures; \( w_N, w_F, w_{\Pi}, \) and \( w_{acc} \) are weights for each metric; and, \( D_N, D_F, D_{\Pi}, \) and \( D_{acc} \) indicate the maximum hypothetical distance between each inputs.

**Definition 10: (Semantic Effect Proximity Measures)**

Given two case sequences \( C_i \) and \( C_j \), and their related term frequency vectors \( \bar{N}_{C_i} \) and \( \bar{N}_{C_j} \) let \( z = |\bar{N}_{C_i}| \) (note that \( |\bar{N}_{C_i}| = |\bar{N}_{C_j}| \)):

\[
d_{acc}(C_i, C_j) = \frac{\sum_{x=0}^{z} \bar{N}_{x,C_i} \times \bar{N}_{x,C_j}}{\sqrt{\sum_{x=0}^{z} (\bar{N}_{x,C_i})^2} \times \sqrt{\sum_{x=0}^{z} (\bar{N}_{x,C_j})^2}}
\]

In addition, cost measures could also be incorporated into our calculation of proximity in order to incorporate the cost associated with making changes to either antecedent of an integration. This can be done by substituting QoS measures and associated accumulation functions with effect scenarios. A weighted sum can be used to appropriately weight each for the comparison.

### 4.4.1 Case Sequence Integration Criteria

Any approach to integration should view both the state [141], the domain knowledge [58] [18] [131], as well co-ordination characteristics [91] [86]. Under our integration scheme we provide a framework for integration based on structural and semantic descriptions. This framework may work in combination with one of the aforementioned approaches.

**Definition 11: (Case Sequence Integration)**

An Case Sequence \( C \) represents the integration of an Case Sequence \( C_i \) and Case Sequence \( C_j \) iff all of the following hold:
4.4. Semantic Process Net Integration

1. Case Sequence $C$ achieves $G_C$ (the goals associated with $C$) where $G_C = G_{C_i} \land G_{C_j}$ and $G_{C_i}$ is the goal achieved by case sequence $C_i$;

2. there exists no $C'$ such that $C'$ achieves $G_C$ and the following holds: $d(C_i, C') + d(C_j, C') < d(C_i, C) + d(C_j, C)$. Here $d$ is a distance function between two case sequences. This defines an integration solution where the closest integration super case sequence has no closer potential solution $C'$.

Note, that our definition of goals above applies to both the functional and non-functional properties of a case sequence.

4.4.2 Case Sequence Integration Methodology

Integration in practice requires some effort on behalf of an analyst, during both process matching and selection of candidate integrations. The criteria we have outlined in the previous sections allow us to reduce analyst effort during the matching and selection steps.

Step 1: Business Process Matching.

Prior to and during integration, matching is required to determine the likelihood that two business processes, or activities within a business process, share similarities or are equivalent. This may involve the use of three techniques. The first involves evaluating the labels of business processes and activities using linguistic techniques (mediated with an ontology) as in [55]. This may help in, for example, determining that a Package Consignment process is semantically similar to a Package Receiving process. Another technique that may be applied (also in combination with an ontology) is the evaluation of the effect (or functionality) of a business process or activity. Here, the semantic aspect to our proximity measure can be re-used effectively. Finally, as processes may be represented at varying levels of abstraction, we can apply the aforementioned techniques to detect the part-whole relations among and within business processes in order to initially resolve abstraction conflicts. These three approaches to matching may be either completely automated, involve some automation, or be a simple guide applied to a completely manual integration.

Step 2: Business Process Integration Goals.

We firstly require a goal (describing a set of criteria) for the integrated business process to be determined. In this approach, the goal can be either given or determined by merging the effect scenarios of each business process to be integrated using
4.5 Summary and Conclusions

an automated or semi-automated strategy. An automated strategy might involve: conjoining consistent effect scenarios; and/or, extracting the most common effects among effect scenarios. As these strategies only lead to some approximate baseline, analysts will need to provide input. The requirement to firstly establish the common business goal for an integration step allows us to reduce the complexity of ad-hoc integration, as well as separating concerns and roles during the process. As discussed, the integration goal can be either computed in a bottom-up or top-down manner, and provides a concise description of the requirements for the integrated model.

**Step 3: Business Process Integration.**

Business process integration involves a search through a space of possible integration options that is directed by our integration characteristics. One way to search for the most proximally efficient integration, can be to follow a local generate and test strategy. Consider an algorithm sketch: whose input is \( \mathcal{P} \) (a process repository to be integrated); and, manipulates a set \( V \) of \( \langle C, \textit{history} \rangle \) pairs. The algorithm would 1: \( V = \{ \langle C, \langle \rangle \rangle \} \) (initialize with the model to be manipulated (possibly the intersection of nodes and edges among models); 2: While(!Accepted(\( V \))) \( V = \text{Step}(\( V, \mathcal{P} \)) \) (step through changes until an acceptable integration is identified). An implementation of the \( \text{Step} \) function would apply a single admissible addition or removal of a node or edge (possibly from elements of \( \mathcal{P} \)). The history would allow: poor candidates to be forgotten; ensure complementary operations are not applied to single models; ensure uniqueness is maintained across models; and provide a local basis for evaluating proximity and other heuristics. Firstly, termination could be an issue due to the infinite (gateways) nature of \( \mathcal{P} \), although results are anytime. There is also a large branching factor, although the measures we have defined guide search.

4.5 Summary and Conclusions

The interesting element of our method is in the use of minimal change of processes. This acts in favor of a business implementing a change management solution in terms of costs minimization (as it costs less to change less), and also in the reduction of change risks. This risk is of growing concern for compliance reasons, as with strict regulative control acting on many businesses it is assumed that broad innovative changes to processes as the result of any integration activity may leave an organisation vulnerable to breaks in the value chain or penalties bought about by uncompleted activity steps.
In this chapter, we have presented an innovative method of process integration. Process integration is the activity of consolidating several processes into one. The method presented uses a set of general proximity measures. The proximity measures show similarity across structural, semantic and statistical bounds. Each measure has been described using a general process model definition and as such each can be leveraged in domain specific applications. The method presented provides a means to complete process integration. The presented framework is novel as it aggregates several measures. The framework is general and is extendable through the incorporation of other similary measures. The future direction of this avenue of study is a review of the best similary measures for domain specific integration projects. In the next chapter, we will discuss organisational strategies and business service descriptions. Using the effect accumulation and similarity measures shown in this chapter, we will describe a method for change management based on organisational strategies.
Chapter 5

Strategic Alignment

*The Feynman Problem Solving Algorithm:*
1) Write down the problem.
2) Think very hard.
3) Write down the solution.

**STRATEGIC alignment is a mechanism by which an organisation can visualize the relationship between its business processes and strategies. It enables organisational decision makers to collect meaningful insights based on their current processes. Currently it is difficult to show the sustainability of an organisation and to determine an optimal set of processes that are required for realizing strategies. Further, there is not a general framework for strategic alignment that can ease this problem. In this chapter, we propose such a general framework for strategic alignment, which helps develop a clear understanding of the relationships between strategies and business processes. The framework gives organisations an understanding of the relationship between a set of processes and the realization of a set of strategies; it also shows the optimal set of processes that can achieve these strategies.**

5.1 Introduction

Strategic alignment is a method for understanding the nature of a business through the correlation of business processes and strategies. The use of strategic alignment allows an organisation to contemplate its longevity and to find how achievable its visions for the future are. Within the realm of service oriented architectures, verification and validation are significant areas of study. Finding correlations between strategies and business processes are a key component of any SOA methodology
In this chapter, we build on the foundations of model validation for the description of business process alignment to ensure that there is alignment between processes and strategies. The method of alignment discussed in this chapter will enable organisations to find if they have the right processes to fulfil their strategies; and thus, will form the basis for understanding sustainable businesses. Our framework for alignment follows from the definition of most specification validation problems \[\text{[116]}\] with the extension that we are interested in optimizing the use of processes to fit the given strategies.

During the creation of workflow systems, process designers strive to create process models or designs that can be considered sustainable \[\text{[21,54,21]}\]. The problem for these activities is in defining the meaning of sustainable process designs \[\text{[116,21]}\]. There is a need to describe and to be able to explain why a process model is sustainable and necessary in a given setting \[\text{[115,116]}\]. By process sustainability, we refer to the long-term effectiveness of a business utilizing efficient processes, measurable through the number of strategies that a business is able to enact. Process models can be viewed as sustainable if they realize part of an organisational strategy. Process models are efficient if when used by an organisation they produce optimal results for the organisation based on some quality of service (QoS) measure. organisations are sustainable if all their strategies are realized by a process. The organisational strategy “ensure that employees are happy” and a process designed to make employees happy can be used to illustrate this point. The process would be aligned to the strategy as it realizes the strategy and hence should be considered sustainable. If there are two such processes for making employee’s happy, then the optimal process is the process that satisfies a desired QoS description, such as, make employee’s happy quickly. By identifying the points of interaction between processes and strategies analysts are able to tell if the processes that they have designed are sustainable.

Results from this work hold numerous benefits for designers who ask What? and How? questions, such as What strategy does this process seek to satisfy? and How is this strategy realized? Through the use of the alignment framework presented in this chapter, analysts will be able to describe and explain a specific process model’s sustainability. The framework that we propose also provides a mechanism to compute the most optimal model of alignment, which shows the best way to realize given strategies in an organisation.

The contributions of this chapter are as follows. First, we propose a framework that grants business analysts the ability to correlate processes with strategies. Secondly, we describe how an organisation can find how many of its strategies are real-
5.2 Strategy Modeling

5.2.1 Strategy Modeling Language

When thinking at an organisational level, it is common to focus on the notion of a strategic plan. Generally, a strategic plan can be broken down into the following headings:

- Organisational description that includes its structure, vision and mission statement.
- Context describing the industry scope as trends, advances, opportunities, and weaknesses. The organisational context will also include information about governance and distinctive competencies that make the organisation unique.
- Risks including both new and existing. The strategy should list ways that the organisation has identified to mitigate these risks.
- A strategy model is a hierarchical definition of strategies that the organisation believes will help move it forward while also addressing contextual issues and identified risks. Each strategy usually has an aim, purpose, constraints and stakeholders.
- Processes / Programs are lists of the operational mechanisms that an organisation will use to realize strategies in the strategy model.

1The strategy modeling language and ontology developed from this research can be found at http://www.strategyontology.com/
In previous work [69], we have proposed a language that can be used by senior executives for describing organisational strategies as part of the strategy model in a strategic plan. This language is the strategy modeling language (SML). The core modeling elements of SML are: Functional Goals, Plans, and Optimization Objectives. These elements can be used in combination to describe a typical strategy model.

There has been a vast amount of work describing strategy modeling such as that done in [69]. A strategy model in the strategy modeling language (SML) [69] can be constructed to describe an organisation’s strategies. SML was developed to provide analysts with a crisp language for describing organisation strategies using Goals, and Plans, Optimization Objectives. Using SML we have established an alignment function that maps cumulative effects to strategic goals.

For the most part, goals are the building block of strategy description languages. When describing a goal, various requirements are encoded (as part of the goal description) to the goal $G$. When the goal’s requirements are achieved the goal is realized. Functional Goals describe outcomes that organisations would like to achieve. When written in SML, these can be evaluated as either fulfilled or not fulfilled. Functional goals are used to reflect internal and external realities that an organisation wishes to achieve, and generally address strengths, weaknesses and opportunities that have been identified in a SWOT analysis.

In most strategic plans, it is common to identify functional goals that for the short, medium and long terms. When these goals are explicitly ordered then they become part of a strategic plan. Each plan in SML describes milestones in an organisational strategy. A plan is an ordered sequence of functional goals. Plans may follow tactical decisions that describe a means to realize higher-level goals.

An optimization objective in SML is used to discriminate over preferences for strategic outcomes. An optimization objective is typically either the maximize or minimize of a function on a set of given QoS capabilities.

A strategy $S$ is either a plan $L$ or a functional goal $G$ or an optimization objective $O$. A strategy model $S$ is a set of all strategies that are to be analysed.

### 5.2.2 Strategy Ontology

To develop the strategy ontology we reviewed [42] that documents a collection of business plans detailing operational strategies of companies in the United States. Through this review, we found common patterns and elements in each of the business plans. From this list of patterns the strategy ontology was created. The strategy ontol-
ogy provides a concept hierarchy of elements traditionally found in strategic plans. The concepts in the strategy ontology are broken down into three primary categorizations, including a business assessment and mission statement concepts (under the concept area of the organisation), tactical strategy concepts (under the concept area of strategy), and operationalization concepts. In addition to the major concepts, there was one recurrent supporting top level concepts, the conceptual context.

![Diagram of the strategy ontology]

Figure 5.1: Primary concepts in strategy ontology

The ontology can be used to describe an organisation in both a top-down and bottom-up manner. In the first instance by using the organisational strategy and the context concepts one can describe the organisational vision and abstract strategies. In the second instances, process and process context can be completed to define a strategic context. For each, risk is modeled at the level of abstraction being described.

5.2.3 Organisational

To describe an organisation the concept tree starts with the organisational industry, customers and mission statement. The mission statement is built from answering key existential questions about the organisation reason for being. The mission statement is also used to describe information about the organisation’s philosophy and direction. To elicit a short-term and long-term mission statement a serious of concepts can be used including:

- Key Industry – This is the description of the primary industry that an organisation operates in. Businesses can operate in multiple industries at the same time, however typically a business does have a major focus area. Secondary
industries should be listed afterward. An industry is defined by the context, customers, industry type and cycles.

- **Key Products/Services and Market** – This is a description of the essential services that a business provides. Again, there may be many services or products that a business produces; however, the key products/services are the focus of the core business activities. The products and service concept are described further in the operational concept.

- **Why does the organisation exist** – Alternatively, “what would change if the organisation closed.” For any business activity, there must be a purpose. During company formation, the forming party usually justifies the reason for the creation of the business. This concept highlights the value proposition that this business brings to this industry.

- **What is unique about this business** – All businesses offer a unique value proposition. Whether it is to meet market demand or where it is to provide high level services.
5.2. Strategy Modeling

- Key customers – Every business produces a product or service for a customer base. Listing a specific key customer demographic is important for any business. This concept is used to identify a customer and potential stakeholders. Listing the key customers enables pinpointing of exactly whom the organisation is targeting with their products and services. A customer concept is defined using the purchase history, budget and demographics.

- Key strategies now and in the future – As the company is constantly evolving its strategies will also continually change. A mission statement, describes the purpose for existence and strategies are the way by which a business attempts to satisfy the mission statement. This concept will strengthen the mission by enforcing documentation of goals for assessment in the future.

- Key issues – This can be elaborated on by providing a risk concept for each issue. Identifying challenges to the strategy enables businesses to create a mitigation strategy. In the original ontology, the key issues were limited to major political and management issues, and strategies for measuring and mitigation. This concept should include an overview of what the organisation is trying to achieve. It should describe the current issues for the industry segment, which can be used to identify weak areas or opportunities for the business for targeting and improvement.

As part of the organisational concept hierarchy, a concept node for describing the industry is included; where details of both primary and secondary industries for the organisation can be recorded. For each industry type, a new subsection can be created. Each industry concept should include details on the type of industry and that part of the industry that the business operates in. Industry type is used to describe the abstract type of industry such as manufacturing, health or logistics, etc. Decomposing industry into segment allows businesses to identify customers and competitors in their niche operating segments. An example of a segment of the logistics industry is freight couriers. Given the industry type definition, industry cycles must also be defined. This allows an organisation to list details on industry specific phrases.

- Each industry may go through many phases, for example, in logistics phases can be seasonal, in education phases can be demand driven.

- After listing all phases that exist for the particular organisation, the current phase can be described in the current phase concept. Including details on when the phase will end and rationale for any assumptions.
Within the industry definition, there is a placeholder for an expanded context concept. Context is discussed in the following subsections as it is a top level concept in the ontology.

Finally, when describing an industry, there is be a description of customers for that industry. This means describing industry wide customers and not just customers of the organisation. The simplest method for doing this is by using key demographics of industry customers.

5.2.4 Strategy

![Strategy Concept Diagram](image)

Figure 5.3: Strategic concept in Strategy Ontology

The business strategy can be described by using the strategy concept hierarchy. Firstly, the concepts of strategy type can be elaborated to optimization objective, functional goal, and plan. As strategies can be defined by a hierarchical structure, there is a parent, sibling and children concepts (under hierarchy).

Each strategy must have an associated rationale which is a description of the purpose, constraints and various stakeholders involved with the strategy.
5.2. Strategy Modeling

Strategic purpose provides the rationalization for incorporating the strategy into organisational desires. The purpose includes details about the ramifications for the organisation if the strategy is satisfied. Details on ramification of failure of the strategy is included to add to the value concept also described in the strategic purpose. With any strategy there should be an evaluation criteria defined. This is dependent on the type of strategy, define; however, should also include metric test and measures of realization and effectiveness.

We elaborate on the strategy concept types further in the section on strategy modeling language.

5.2.5 Operational

Strategy operationalism, is a general concept used to describe actionable elements of an organisation at a high level. Broadly, operations can be described through assets, opportunities, process, products and services and quality. To describe operational assets, each resource can be expanded on. For any given resource, there is a need for governance as well as details on current stocks, projection stocks, and supply. Opportunities are used to describe the customer pipeline and operational measures that apply to the supply of services to current customers. These measures can be in
the form of capacity and utilization. In addition to current opportunities, the concept can be expanded with details on local, national and global growth and market share.

![Diagram of Products and Services expansion in Strategy Ontology](image)

Figure 5.5: Products and Services expansion in Strategy Ontology

Importantly, within the operational concept, there is a space for details on products and services. Each product or service can be defined by including growth strategies, details on margins, returns, turnover, and delivery. Which can be refined further with a context that includes details of distinctive competencies.

Finally, operations include both process and quality concepts. The process provides a language for describing level 1 processes at a broad level, and quality provides a language for describing operational qualities in terms of production and market based quality measures.

### 5.2.6 Context

The context concept is a large category concept, developed to provide a common language between operational and organisational concepts in the strategy ontology. Broadly, the context of any concept in the ontology includes a language for describing:

- **Element names** – An element name is a name given to a strategy, process, or other concept in the strategy. Each name is describable using a textual label, version, and description.
5.2. Strategy Modeling

- Governance – Governance describes the organisational rules for strategy and process definition, update, and removal. Governance is used to describe information about ownership, priorities and permissions. Within the concept in our strategy ontology, governance can be described with information about consumer, owner, requirements and supplier.

- Distinctive Competencies – is used to describe and to link products and services to strategic concepts and organisation capabilities. Each competency can be described with a name, evaluation criteria, priority and key products / services.
5.2. Strategy Modeling

- Risks – Risks are a large sub-context concept and during the development of the ontology it was debated as to whether Risk should be its own top level concept. It has ultimately been included as a contextual description as it can be used to describe any strategy context. Risk is generalised as a classification which is described by entry and exit barriers, issues, probability of occurring, type, time period and various risk factors.
• Scope – The scope concept leans towards being operationally orientated, though does provide links back to the general strategy. Scope can be used to describe constraints, state and stakeholders as well as various risks associated to a given context. Each scope can extended with detailed opportunities for the market in a given timescale.

• Value – Value is left as an abstract concept in our strategy ontology, as it can have many different meanings at different levels of abstraction.

5.3 Strategic Alignment of Business Processes

In this article, we will consider strategic alignment using a notion of process composition. This concept is required for describing business process alignment, as normally, we have found that business process models typically do not realize strategies by themselves, because strategies are described in more general language than process models. For example, a business process that describes a set of activities for evacuating a building will not necessarily satisfy an organisational goal to ensure that employees are safe. In our framework we leverage a composition of processes that contribute to the safety strategy of the organisation. There is a general need within businesses to connect similar processes and services that meet the needs and demands of different functional requirements [59]. Processes can be composed using either parallel or sequential process semantics, where the parallel joins have corresponding semantics to a BPMN AND gateway. A sequential composition has similar semantics to sequential activities within BPMN joined by sequence links.

When discussing process models, we refer to a process portfolio [178] as an organisation-wide collection of business process models. Each process in a process portfolio describes the capabilities and activities involved in the execution of each process model. Given our description of process effect accumulation, we will be considering alignment between single processes and strategies, as well as alignment between composed processes and strategies. Given a process portfolio $\mathcal{P}$, we shall use the term composite process portfolio, denoted by $\mathcal{C}_P$, to describe the set of all possible compositions of processes in $\mathcal{P}$.

**Definition 12 (Realization):**

A process $P$ with a set of end effect scenarios $E_P$, realizes a goal $G$, if and only if an end effect
scenario of $P$ entails $G$, i.e., $\exists \epsilon \in E_P \text{ s.t. } \epsilon \models G$. We will write: $P \text{ alignedTo } G$ if this is the case.

Consider a set of process models $\{P_1, \ldots, P_n\}$ in a process portfolio $\mathcal{P}$. Given a goal $G$, we want to determine if the process portfolio $\mathcal{P}$ is aligned to the goal $G$. Trivially, we have the following basic test: if $\exists P \in \mathcal{P}$ s.t. $P \text{ alignedTo } G$ then, $\mathcal{P}$ is aligned to $G$; however, we also need to consider the possible compositions of the processes in $\mathcal{P}$. If a goal can be realized by a process in the composite process portfolio then the process portfolio is aligned to the goal.

**Definition 13 (Alignment with Goals):**

Let $\mathcal{P}$ be a process portfolio, let $\mathcal{C}_P$ be the composite process portfolio derived from $\mathcal{P}$ and let $\mathcal{G}$ be a set of goals. $\mathcal{P}$ is aligned to a single goal $G$ iff $\exists P \in \mathcal{C}_P$ s.t. $P \text{ alignedTo } G$. This is denoted $\mathcal{P} \text{ alignedTo } G$. We will say $P \text{ alignedTo } G$ iff $\forall G \in \mathcal{G}. P \text{ alignedTo } G$.

Strategic plans are sequences of strategic goals (or other plans). Each plan describes milestones in an organisational strategy model. Where each goal in the sequence must be achieved before its successor goal. A plan in a strategy model is a temporal sequence of goals.

**Definition 14 (Alignment with Plans):**

Let a plan $L$ be a sequence of goals $\langle G_1, \ldots, G_n \rangle$. For the plan to be completely realized by a process model (or process models) each pair of consecutive goals $\langle G_i, G_j \rangle$ in the plan must be realized. A plan is realized and aligned to a set of processes if all consecutive goal pairs in the plan are realized. Pairs of goals are realizable in the following ways:

1. Given two processes $P_k$ and $P_l$, where the processes can be composed in the sequence $\langle P_k, P_l \rangle$ to form process $P_m$, if $P_k$ realizes $G_i$ (but not $G_j$) and $P_m$ realizes $G_i \land G_j$ then the process composition $P_m$ realizes the goal pair.

2. Given a semantically annotated process model $P_n$, where there is an activity $a$ with effect scenario $\epsilon_a$ that entails the goal $G_i$ and there is an activity $b$ with effect scenario $\epsilon_b$, that occurs in the pathway after activity $a$, that entails $G_i \land G_j$ and there is an end effect scenario of process $P_n$ that entails $G_i \land G_j$ then the process $P_n$ realizes the goal pair. The effect scenario $\epsilon_a$ must not entail $G_i \land G_j$, otherwise the realization order of the goals will be incorrect.

To compute optimization objective Alignment, given a strategy $G$, and two processes $P$ and $P'$ with alignment relationships $P \text{ alignedTo } G$ and $P' \text{ alignedTo } G$. We need to add a mechanism for determining which process is a better fit for the strategy. To
5.3. Strategic Alignment of Business Processes

do this, we refer to a process capability function that computes the value for processes satisfying a particular strategy; similar functions and capabilities are shown in [159]. The function will return the best process from a collection of processes that can satisfy the strategy with a given objective function. For example, consider an organisational optimization objective $O : \text{`minimize cycle time'}$ applied to a functional goal encouraging the use of vacation time. Process $P$ may be a manual process that requires the employee to submit leave request forms and find their own replacements, and process $P'$ may be an automated process that automatically selects replacement employees and streamlines the approval process. A QoS execution description for process $P$ may be $\text{Time < 2 days}$, and the QoS execution description for process $P'$ may be $\text{Time < 2 hours}$. Provided that there are no alternative QoS objectives, then the selection function will select process $P'$ as being the optimal process to satisfy the goal.

**Definition 15** (Alignment with Optimization Objectives):

Given a strategy $G$, an optimization objective $O$, and two realization scenarios $P \text{alignedTo} G$ and $P' \text{alignedTo} G$, then we refer to the optimal candidate process for realization as the most optimally aligned process $P \text{alignedOptimallyTo} G$ which is the process that is more preferred based on the optimization objective, i.e. $P \leq O P'$ iff $P$, satisfies the optimization objective $O$ better than $P'$. Similarly, if for a strategy $G$, there is a set of processes in a process portfolio $P$ that optimally realizes the strategy, then the realization is denoted $P \text{alignedOptimallyTo} G$.

We observe that this selection of optimal processes has been discussed in other research such as in [207]. Using the previous definitions of goal alignment, plan alignment and optimization objective alignment, we can now tie together an alignment definition that can be used to describe strategic business process alignment.

**Definition 16** (Strategic Alignment):

Let $C_P$ be the set of the composed of processes of $P$. Let $G$ be a collection of strategies. $P \text{alignedOptimallyTo} G$ iff:

1. For each $G \in G$: (completeness)

   (a) $\exists P' \subseteq C_P. P' \text{alignedOptimallyTo} G$

   (b) There is no $P'' \subset P'$ where $P''$ satisfies condition a. (realization minimality)

   (c) $\neg \exists P \in C_P. (P \land G \models \bot)$ (consistency)

2. There is no $P^* \subset P$ where $C_{P^*}$ satisfies condition 1. (alignment minimality)

It should be noticeable that there are differences in purpose between Definition 16 and Definition 13 as minimality conditions are missing from Definition 13. In this
setting we argue that finding the best set of processes that are able to meet an entire organisational strategy is of great importance for both executives and analysts.

We will now step through an example of alignment between an organisation’s business processes and strategies.

**Example 1** (Strategic Alignment Example):
Recall from the motivating example §3.3 there are a number of strategic goals to be realized. In the process portfolio, a number of processes are available for analysis to test if they can be utilized in optimal realization of the organisational strategies. For each process there is a QoS metric for `TimeTaken` annotated at the bottom right of each model.

We must ensure that all strategic goals are realized by the processes in our process portfolio.

First consider the goal: Encourage the use of an employee’s holiday period → `HolidayProvisioned` with the optimization objective minimize wait time for holiday approval, to which the Employee Vacation Request is aligned as the effects of this process realize the goal condition and the QoS variable for Employee Vacation Request is minimal compared to the alternative goal realizing process Manual Employee Vacation Request.

Next, we consider the goal: Maintain retention of high-quality staff → `EmployeeRetention ∧ HighlySkilledWorker`. From the knowledge base EmployeeRetention is achieved if there are processes that ensure `HappyEmployee ∧ SalaryPaid`.

The Salary Payment Process has the effect SalaryPaid and the Employee Vacation Request process ensures that the effect HappyEmployee is fulfilled. Training Process has the effect HighlySkilledWorker which completes the requirements for the goal to be realized.

Finally, we have a plan with two subgoals: Ensure that staff are the best in the industry. To realize this plan, we construct a composed process where we attempt to satisfy each goal.

For the sub-goal: Maintain ongoing training → `TrainingProvided` the Training Process is aligned as the effects of this process realize the goal condition.

The sub-goal: Maintain high employee morale → `HappyEmployee` can be aligned with Employee Vacation Request as the effects of this process realize the goal condition.

The process composed of the Training Process and the Employee Vacation request realizes the plan.

On review of this example, we can determine there is no smaller set of processes we could use to realize all the organisational goals from the example case. As a final analysis on this alignment, we can suggest to the organisation that it could drop the manual employee vacation request process.
5.4 Implementation

To demonstrate the use of our framework we have sought to extend the functionality of Process Seer through a text based toolkit without BPMN modeler support. The tool is discussed in detail in Chapter. Currently the tool is able to load and test process models for consistency against a rule base. The tool builds sequential and parallel process compositions, then ProcessSeer style effect accumulation can be computed on the composed process models to find composition end effect scenarios.

Figure 5.9: Implemented Tool TextSeer

Although we have based the tool on ProcessSeer, the accumulation engine is more extensible. In ProcessSeer accumulation is done in a single mode of belief update. This is due to the algorithm design and the accumulation function. In text seer we have implemented the same base accumulation algorithm with the added benefit of deductive closure at each pairwise accumulation step; additionally, a default logic reasoner has also been implemented such that the accumulation function could be replaced with a default logic reasoner to provide belief revision style accumulation during the process effect accumulation.

Depending on the application of accumulation either a belief update or a belief revision function can be used, in future work we aim to investigate the benefits of

\footnote{For source and further implementation details see \url{http://www.fnord.be/textseer}}
each style of accumulation for differing domains. For example, in a customer domain a belief revision function may be more effective as the process may wish to retain knowledge of previous interactions and update its knowledge base accordingly. In the domain of a traffic control process system belief update may be better suited as updates force the removal of bad data.

To test the effectiveness of our tool a randomised process algorithm was also developed. The algorithm produces process models and effects and then attempts to accumulate the effects. This is done to assess the computational time that would be required to conduct alignment across a large amount of organisational data. As expected when the number of processes and effects scenarios increase the computational time needed to compute and effects scenarios grows exponentially.

5.5 Conclusion

In this chapter, we have described alignment as a realization relation between a set of process models and a set of strategies. We have presented a formal framework that can be used to show optimal strategic alignment within an organisational context. The framework contains a set of methods for correlating process models to functional goals, strategic plans and optimization objectives. Further, the result of using strategic alignment to determine the alignment between strategies and processes shows an organisation the optimal set of processes from a process portfolio that would help them realize their strategies. The framework that we have presented contributes to a better understanding of strategic business process alignment, and further tool support will equip decision makers with a device to understand sustainability in an operational context. In future work, we will look to extending the definitions of strategic alignment and then show a method for discovering business processes that can meet organisational strategies, attempting to provide a method for rapid business infrastructure development.
Chapter 6

Organisational Compliance Management

But of number, cosa, and cubo, however they are compounded...,
When in your equations you find terms with different intervals...
You shall say that the art, has not given the solutions to this case...
even if the case may be possible.

LUCA PACIOLI

Business process compliance management is a field of study involving the coordination of business process management and compliance systems. We define a compliance system as an organisation wide tool that links legislative and business rules to organisation policies and processes. The objective of such a system is to promote a self sustaining level of operations that minimizes the losses caused to the business through breaches of laws or internal misappropriations. We argue that it is also possible to view a compliance system in a similar fashion to that of an accounting system where each process is treated as a transaction. Each process may be monitored through logs of multiple instances of execution. The result of which can be evaluated for costs and benefits; allowing a utility to be associated to each activity within the process. By assessing both the high order policy creation within a company as well as low order transactional histories of single processes a complete picture of current operations can be obtained. In this chapter we discuss benefits and shortcomings in existing compliance schemes and present a our own method for measuring the degree of compliance that each business process may achieve.
6.1 Introduction

A compliance system is an organisation wide tool that links legislative and business rules to organisation policies and processes. The objective of such a system is to promote a self sustaining level of operations that minimizes the losses caused to the business through breaches of laws or internal misappropriations [16,52,106,146,193]. Compliance management is a growing concern for organisations. Each organisation is in a constant battle between: the forces of market demands, shareholder ROI, sustainability measures to ensure the future viability of of itself (as an entity) as well as the environment (as a collective), and the legislative requirements imposed by continuously increasing laws.

There is a low consumer confidence in the public sector and it is believed that reportedly healthy public companies are vulnerable to multibillion dollar restatements due to non-compliance bought about by internal and external pressures. There are concerns in the market that further government regulations will lower confidence further [9,56]. Conversely it would appear (from recent trends) that regulations through heavy putative measures are the only effective answer. We propose that new legislative efforts are not necessarily required. Instead we aim to show how a more holistic approach to compliance management that provides a more rigorous method for compliance checking may be of benefit. The auditing profession is seen to be dissolute as well, with accounting companies taking equity holdings in their customers as compensation for their creative accounting. It has been hinted that even basic principles like GAAP (generally accepted accounting principles) are being ignored or used to cloak these fraudulent activities. “By 2002, the US General Accounting Office reported that 10% of all listed companies restated their financial results in the period 1997-2002. In 2001 alone, there were 225 restatements. [According to the US general accounting office (GAO), in a landmark study in 2002 (US senate 2002), it was found that financial restatements increased significantly from 92 in 1997 to 225 in 2001and increase of approx 145% ]” [139]. Other motivations for adopting a compliance framework are outlined in [3,52,56,106,139]. These include: Increasing threats to information security, alignment of technology projects with strategic goals to ensure maximal value addition to modern product lines, and the increased risks of intellectual property destruction by atomic service creation.

To construct a compliance system, requirements should be first identified and then mapped against the business process. This mapping involves associating degrees of compliance with each requirement and then using the degree of compliance to
determine the level of compliance of each execution. A compliance system may then be used to measure the ‘degree’ of compliance for a process execution.

We believe that a fundamental basis for assessing effective compliance as a complete system is needed, rather than creation of independent monitoring functions and varying policy recommendations for each new law that is enacted. We have developed a general framework for compliance management using c-semirings [20] to aid in the dissemination of feedback on the operational compliance for various organisational processes. This feedback may be used by law makers, law enforcers, compliance enforcers, or organisational compliance committee’s. We have also used lattices [31,53,103] to aid in the creation of viable process compliance rankings.

Using the combination of c-semirings and lattices we aim to answer the following questions, if there is a case that a company must comply with legislation, how does it achieve compliance at a minimal cost for maximum benefit? Is there a method to implement a broad measurement device for satisfying compliance requirements? And argue that in order to understand compliance one must view both the high order policy creation (Design Time Compliance) as well as the low order transactional histories of single processes (Run Time Compliance) to get a complete picture of current operations.

To many parties compliance means many things, and in the broadest sense “Compliance is about meeting particular acknowledged obligations that may have a mandatory component to them”, by “creating management systems and operational procedures and ensuring ongoing monitoring” [40].

**Components of a System**

The primary actors in any compliance system include the regulators, the regulated and the public. Regulatory bodies are compliance enforcers. Receiving new laws from legislators they adopt the implementation policies and monitoring procedures to verify compliance with requirements from constituents. Regulators provide clarification and feedback of current practices as a function of their duties. Public companies are the implementers of the laws. Each must ensure that current operations are undertaken in a legal fashion in order to guarantee risks are minimized. Monitoring and feedback are essential to implementation as a means to verify organisational compliance. Stakeholders are the beneficiaries of compliance. Laws are enacted to protect financial investments and public safety. The role of a beneficiary is to provide feedback on the effects of compliance, unwittingly in most cases the stakeholders
suffer passed down burden of cost of compliance. Compliance is a chain of creation, institutionalization, implementation, evaluation and feedback return [52,139,188].

A process is a set of activities that are completed in order to meet various business strategies. For example, if the goal is to create a new user on a computer system then a process is a set of the activities that can be completed for the fulfilment of the goal. Each process can also be abstracted into a single activity over a larger process. When referring to a process instance, we refer to a single execution of the selected process. A compliance requirement is a measurement function that returns a value of performance based on the process instance that has just completed.

A criterion in building a generalized compliance system is research into the importance of continuous automatic assessment of general process level transactions. This along with the inspection and real-time processing of transaction logs to reinforce the effectiveness of the total control environment appears to be lacking from the previously described general assessment frameworks. The advantages of automated transaction monitoring are that they can provide a more visually understandable view of actual processes [16,138]. They can also increase trust of an organisation from a stakeholder position as a business is much more transparent [138].

 Compliance in terms of design is heavily influenced by standards within most corporate governance frameworks. A compliant organisation aims to minimize risks through use of five key principles [16]:

1. Predictability - Regulations within a stable system for creating and enforcing policies. If an organisation is consistent with policies that work, then that organisation lowers risks. There is a question of reliability in predictability [119] and propensity to change [47].

2. Transparency - Clarity and availability of information. If there exists a clear framework and clearly defined policies then it follows that any person or stakeholder can see the chain of reporting, clear rule definitions, and identify accountable parties. We provide a case study to show the disparity in providing information across a population and the importance of transparent communications [25,26].

3. Accountability - Repudiation through segregation of duties is a desirable element in any system. It should be clear that those who control and manage companies must be recognizable for when and how their activities breach pre-
6.2 Motivations

Compliance requirements and rules can be broken into contractual rules \cite{82} that can be accumulated across business process models \cite{133}. The problem that occurs is when requirement values are not so clear. An imprecise compliance requirement is a requirement that falls between crisp true and false values. In this section we will define a spectrum of crisp and imprecise compliance requirements.

In an attempt to measure the level of compliance of a business sector or complete organisation, we are faced with the dilemma of determining the degree of compliance for a processes. This is a relatively straightforward task for crisp compliance requirements, as it is obvious to say ‘Yes’ the process met the requirements, or ‘No’ the process did not meet the requirements. The separation of crisp compliance requirements produces a Boolean value that can be used in the analysis of requirements \cite{14,223}. The problem that occurs is when requirement values are not so clear. An imprecise compliance requirement is a requirement that falls between crisp true and false values.
6.2. Motivations

We need to be able to determine, for each process instance, and for each compliance requirement, the “degree of compliance” of that instance with that requirement. For crisp compliance requirements, the assessment of degree of compliance is Boolean, i.e. a process instance either does or doesn’t comply. For more imprecise or vague compliance requirements, the assessment involves greater complexity. Consider a compliance requirement that states: “quarterly activity statements must be filed within a reasonable time frame”. Clearly an activity statement filed immediately after the end of a quarter satisfies the requirement entirely, while one that is never filed violates it entirely. A statement filed 10 weeks after the end of the quarter satisfies the requirement partially. A statement filed 12 weeks after the end of the quarter also satisfies the requirement partially, but to a lesser degree than the statement filed 10 weeks after the end of the quarter. A mechanism for assessing degrees of compliance that sit between the two extremes of full and partial compliance is therefore required. If we consider that ‘Statement lodgement in under 4 weeks’ to be a satisfactory achievement of compliance, and that ‘Statement lodgement in under 2 weeks’ is a good achievement, and ‘Statement lodgement in under 30minutes’ to be a great achievement we can start to categorize various processes with a degree of compliance. We call the degree of how a process satisfies its compliance requirement the degree of compliance of a process.

Example of a crisp compliance requirement (from [10]): “When a negative statement is made and a new debt is to be incurred, the directors should have regard to s592 of the Law which makes it an offence for a director to allow a company to incur a debt when at the time there are reasonable grounds to expect that the company will not be able to pay all its debts as and when they fall due. The directors are also jointly and severally liable for the payment of the debt.”

An imprecise compliance requirement is not always clearly defined and may be interpreted differently by different parties.

Automation and the Role of Continuous Monitoring

A criterion in building a generalized compliance system is research into the importance of continuous automatic assessment of general process based transactions, value addition and value losses. This, along with the inspection and real time processing of transaction logs to reinforce the effectiveness of the total control environment, appears to be lacking a general assessment model, and instead implementations appear to be adopting specialized systems that may be incompatible with policy based
6.2. Motivations

frameworks. The advantages of automated transaction monitoring are that they can provide a more visually understandable view of actual processes. They can also increase trust of an organisation from a stakeholder position as the business is much more transparent.

This transparency can be seen in the enterprise architecture developed for the USCP Legislative Branch \[67\] where value metrics were used to determine risk factors for activities. In knowing each risk involved the USCP was able to recreate better policies and streamline their operations.

We’ve found that any compliance system requires both a design time compliance model to be put in place alongside a complementing Run Time compliance tool that should allow a company to monitor and review ongoing transactions with much more rigour.

Differences between Run-time and Design-Time Compliance Systems

A unifying schema for assessing the overall tone of corporate compliance requirements has not been invented; which, may play a part in the genesis of current fears against public companies \[9,56,139,188\]. The majority of current frameworks does not provide a measurement of the effectiveness in a broad set of an organisation’s compliance. This is because the existing theoretical accounts are not created to monitor both run time and design time systems across the full system. General research into the field of compliance has taken on a very narrow perspective. Existing frameworks approach compliance as a purpose of policies or as a function of monitoring rather than both. There are calls for investigations of combined systems \[52,119,188\] that should show a breakdown of compliance functions across a complete system.

<table>
<thead>
<tr>
<th>Design Time</th>
<th>Run Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Governance Policies</td>
<td>Degree of Compliance</td>
</tr>
<tr>
<td>Role Model, Segregation of Duties</td>
<td>Audit Results</td>
</tr>
<tr>
<td>Meta-Process Statements</td>
<td>Obligation fulfilment</td>
</tr>
</tbody>
</table>

Table 6.1: Compliance RunLevel Comparison

**Meta-Process Statements**: Meta process statements define various abstract processes inside an organisation. They are not defined rigidly as obligatory processes. Meta-process statements may be considered possible unconnected business process
components. Meta-process statements are imperial policies that give an actor a choice of actions that they can use and combine to complete the task at hand \[52,139\].

**Governance Policies:** Compliance rules and governance policies define requirements on the job itself. The importance in the foundation of governance systems is the inclusion of penalties for non-conformity \[81–83,133,193\]. Governance policies must be taken as authority policies that provide an actor within the organisational authority power to execute their function.

**Role Model, Segregation of Duties:** At the most fundamental level, segregation of duties means that no individual should have control of two phases or more than one transaction of an operation. It is one of the most basic internal control mechanisms within any line of work. Crisp segregation \[13\] offers the benefit of fraud detection. In most cases, the fraud requires collusion by two or more people. It is also a lot more probable than accidental errors are found \[65,106,187\]. There is also currently work in deriving individual activities from collective obligations with deontic logic \[38,37\].

**Audit Results:** As standard practice, all public companies must have some level of auditing \[52,65,106,195\]. These offer an abstract understanding of the current organisational state. The major concern with auditing practices tends to be associated with lightened auditing principles. Any system that has a heavy reliance on polite and non-intrusive practices opens itself to hidden fraud \[119,187\]. Transactional data is of great importance within any organisation, as it allows analysis to be performed to provide precise information about the current operations of the business. These results carry vastly improved confidence in financial reports and ratios, and make progress by reducing the maintenance costs of manual compliance and providing a degree of risk management \[65,187,188\].

**Degree of Compliance:** This is the degree of how a business process satisfies its governance policy. This is a performance measure placed on transactional analysis that places the current operational compliance against the optimal or best practices’ ranking provided by the governance policies. There is work in where a Formal Contract language (FCL) \[133\] is applied to map requirements to activities. Each rule in FCL may be employed to offer a degree of single activity compliance. This compliance degree is then accumulated across a business process model to provide a compliance measurement on activities.

**Obligation fulfilment:** During a business process, obligation fulfilment of duties is the obligations of an actor to execute the required duties \[52,106,193\]. In our context, we refer to the fulfillment of contractual duties at a process level. Duty of fulfillment of contracts is the product of applying governance policies to meta-process
6.2. Motivations

statements within an organisational function.

6.2.1 Running Example

For all of the examples shown we will use two processes. These processes have been modeling in BPMN in Figure 6.1 and Figure 6.2.

**Figure 6.1: User Creation Example**

1. The “setting up new user account” process.
2. The “lodgment of quarterly activity report” process.

The **setting up new user account** process has the following requirements:

1. “Selection of a valid username” - A new user must not be allocated a username that currently exists in the username data store.

2. “Selection of a ‘good’ password” - The password must meet the requirements of a ‘good’ password. This may be an imprecise requirement if ‘good’ is not defined.
3. “The user management system must be updated to include the new user”

The *lodgment of quarterly activity report* process has the following requirements:

1. “A quarterly report is to be filed within a reasonable amount of time” - This is a KPI based requirement that is set to limit delays in processing. This may be an imprecise requirement if ‘reasonable’ is not defined.

2. “A quarterly activity report must be typed using the company report template”
   - This is a crisp restriction to the material used to produce the report.

3. “A quarterly activity report must be printed and mailed to the regulatory body”
   - This is a crisp communication requirement, indicating email is not to be used.

### 6.3 Measuring Degrees of Compliance

In this section, we shall define a means for measuring degrees of compliance. Effectively, we wish to understand the degree to which the processes of an organisation complied with the applicable set of compliance requirements, over a given audit period. We propose to use a simple algebraic mechanism for specifying degrees of compliance - the framework of c-semirings [20]. This framework is particularly useful because it permits us to combine assessments on multiple dimensions and on a mixture of qualitative and quantitative rules. C-semirings have been used to formalize soft constraint problems [97] and [107] where different tuples in a constraint satisfy the constraint to varying degrees.

**Definition 17: c-semiring [20]**

A c-semiring is a tuple $\langle A, +, \times, 0, 1 \rangle$ such that:

(i) $A$ is a set and $0, 1 \in A$

(ii) $+$ is defined over (possibly infinite) sets of elements of $A$ as follows:

- for all $a \in A$, $\sum(a) = a$;
- $\sum(\emptyset) = 0$ and $\sum(A) = 1$
- $\sum(\bigcup A_i, i \in I) = \sum(\sum(A_i), i \in I)$ for all sets of indices $I$ (flattening property);

(iii) $\times$ is a commutative, associative, and binary operation such that $1$ is its unit element and $0$ is its absorbing element;
6.3. Measuring Degrees of Compliance

\[(iv) \times \text{distributes over} + \text{ (i.e. for any } a \in A \text{ and } B \subseteq A, a \times \sum(B) = \sum(a \times b, b \in B)). \]

Each c-semiring has a partial order \( \leq_s \) over the set of values \( A \) where \( a \leq_s b \) implies \( a + b = b \) in c-semiring \( S \).

Bistarelli has provided a number of c-semiring instances in [20], these include:

- **Boolean**: \( \langle \{ T, F \}, \lor, \land, F, T \rangle \)
- **Fuzzy**: \( \langle [0,1], \text{max}, \text{min}, 0,1 \rangle \)
- **Cost/time**: \( \langle \mathbb{Z}^+, \text{min}, +, +\infty, 0 \rangle \)
- **Quality**: \( \langle \{ \text{HIGH, MEDIUM, LOW} \}, \oplus, \otimes, \text{LOW, HIGH} \rangle \) with a partial order defined as:
  
  - \( \text{HIGH} \oplus \text{MEDIUM} = \text{HIGH} \)
  - \( \text{HIGH} \oplus \text{LOW} = \text{HIGH} \)
  - \( \text{HIGH} \oplus \text{HIGH} = \text{HIGH} \)
  
  - \( \text{MEDIUM} \oplus \text{LOW} = \text{LOW} \)
  - \( \text{MEDIUM} \oplus \text{MEDIUM} = \text{MEDIUM} \)
  
  - \( \text{LOW} \oplus \text{LOW} = \text{LOW} \)

An important property of c-semirings is that they can be combined into an aggregate structure. The combination of c-semirings requires a combination operator to be defined for both \( +, \times \) operators.

**Definition 18: Multidimensional c-semiring** [113]

*Let \( A \) and \( B \) be two c-semirings. The operations \( + \) and \( \times \) for the multidimensional c-semirings \( A^n \) produced by combining \( A \) and \( B \) are as follows:

\[
(a_1, \ldots, a_n) + (b_1, \ldots, b_n) = (a_1 + b_1, \ldots, a_n + b_n)
\]

\[
(a_1, \ldots, a_n) \times (b_1, \ldots, b_n) = (a_1 \times b_1, \ldots, a_n \times b_n)
\]

These operations inherit associativity, commutativity and distributivity from the operations in \( A \).*

Using multidimensional c-semirings, we can combine multiple c-semiring instances.
Example 2: (c-semiring Combinations)

Given a Boolean c-semiring $S$ and a fuzzy c-semiring $T$;

$$S = \langle \{ F, T \}, \lor, \land, F, T \rangle$$
$T = \langle \{ x | x \in [0,1] \}, \max, \min, 0, 1 \rangle$

The c-semiring resulting from the combination of $S$ and $T$ can be the set: $\langle T, .75 \rangle; \langle T, 1 \rangle; \langle F, .75 \rangle; \langle F, 1 \rangle$

During the analysis of degree of compliance for a compliance measurement there is a need to identify both the greatest lower bound (glb) of a c-semiring and the least upper bound (lub) of a c-semiring. By identifying the lub and glb of a c-semiring we have a measure of degree of compliance.

**Definition 19: Greatest lower bound, least upper bound [20]**

Consider a partially ordered set $S$ and any subset $I$ of $S$.

- An upper bound (resp., lower bound) of $I$ is any element $x$ such that, for all $y \in I$, $y \leq x$ (resp., $x \leq y$);

- The least upper bound (lub) (resp., greatest lower bound (glb)) of $I$ is an upper bound (resp., lower bound) $x$ of $I$ such that, for any other upper bound (resp., lower bound) $x'$ of $I$, we have that $x \leq x'$ (resp., $x' \leq x$).

A c-semiring can be used to represent all combinations of degree’s of compliance within a compliance system. To map requirements into a form that can be used within a c-semiring instance, we introduce a function describing compliance requirements in the next section and show how such a function may be leveraged to find a compliance measure across an organisation.

### 6.3.1 Compliance Assessment Framework

In this section we first define a compliance requirement that can be used within the c-semiring framework described in the previous section.

**Definition 20: Compliance Requirement**

A compliance requirements is a function:

$$R : 2^\mathcal{L} \rightarrow A$$

where $2^\mathcal{L}$ is the set of all well-formed sentences in the underlying language in which process instances are described. $\mathcal{L}$ is the language in which process instances are described. $A$ is the
set of degrees of compliance associated to the process. R is a degree of compliance measurement function.

Each compliance requirement can be used within an instance of a c-semiring. As an example, a Boolean c-semiring can be used to show how to model simple business process instance.

A process instance can achieve a degree of compliance of either compliant (1) or non-compliant (0); this degree of compliance is determined by whether or not the process has satisfied the requirement. For example:

**Example 3: (c-semiring example)**

Using the process of “Setting up a new user account on the DBMS” from the running example in section 6.2.1 we introduce two instances (each instance will be denoted $P_i$ where $i$ is the instance number) of the process:

- $P_1$, process executed and met the requirements.
- $P_2$, process failed to meet the requirements.

A representation of these process instances modeled in a boolean c-semiring is shown as follows. Let’s consider the following instance of a c-semiring: $\langle \{0,1\}, \lor, \land, 0, 1 \rangle$ where $1$ represents full compliance of a process and $0$ represents a non-compliance process. The operators ($\lor, \land$) are representative of logical or/and functions. The elements being compared are either $1$ or $0$. In two separate instances of execution, the process may adhere to the compliance requirements in the first instance and achieve a compliant degree of $1$. In the second instance of execution the process may achieve a compliant degree of $0$. When these two instances are compared then the $\lor$ operator is used to determine which process instance adheres to the compliance requirements better. When the two instances are combined to find their overall process compliance then the $\land$ operator is used.

A comparison of $P_1$ and $P_2$ would result in $1$ as $1 \lor 0 = 1$. A combination of $P_1$ and $P_2$ would result in $0$ as $1 \land 0 = 0$.

Building on the c-semiring framework we have created a scheme to model business run time compliance by grouping multiple processes instances and their activities that can be tailored to a specific application area for the overall assessment of degree of compliance within an organisation.

**Definition 21: Instance Measure**

For a process $P$, with a set of instances $\{p_1, ..., p_n\}$ over a compliance audit period (this could be the full set of instances over the audit period, or a random sample), the degree of compliance
6.3. Measuring Degrees of Compliance

with \( R \), denoted by \( C_R(P) \) where \( R \) is a compliance requirement, is given by \( R(p_1) \times R(p_2) \times \ldots \times R(p_n) \).

In the following we show an example of this instance measure, extending Example 3 with activities within the process being assessed using the requirements in the running example in Section 6.2.1.

**Example 4: (Instance Measure Example)**

If the process of "creation of a new user on the DBMS" is conducted a number of times in an audit period then a set of instance transaction logs that record how various activities in the process have been completed could be as follows (each instance is noted by \( I_n \)):

<table>
<thead>
<tr>
<th>Transaction Log for creation of a new user on DBMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_1 ) New username: 'John'; New password: 'secret'; SET savedFlag=true;</td>
</tr>
<tr>
<td>( I_2 ) New username: 'Andrew'; New password: 'password'; SET savedFlag=true;</td>
</tr>
<tr>
<td>( I_3 ) New username: 'John'; New password: 'fido'; SET savedFlag=false;</td>
</tr>
</tbody>
</table>

A compliance requirement \( R \) could have associated to it a degree of compliance of 1, 1, 0 for each log entry respectively.

The set of values from the execution instances represented in the log have a degree of compliance with \( R \) that is:

\[
C_R(P) = 1 \times 1 \times 0 = 1 \land 1 \land 0 = 0
\]

Given this degree of compliance with \( R \) a compliance analyst may quickly find non-compliant process instances and fixes to the system can be made.

Various processes may have to conform to multiple requirements. If we consider the aforementioned process as a single process (rather than a collection of instances) then the set of requirements restricting all instances of the process are the requirements for that process.

The degree of compliance of a process with all its requirements can be formulated as follows:

**Definition 22: Requirement Measure**

For a process \( P \), and a set of compliance requirements \( RS = \{r_1, r_2, \ldots, r_n\} \), the degree of
6.3. Measuring Degrees of Compliance

compliance of P with RS, denoted by $C_{RS}^\Sigma (P)$, is given by

$$C_r_1 (P) \times C_r_2 (P) \times C_r_n (P)$$

Next we show the overall measure of compliance by combining multiple requirements onto multiple process instances. This is done by combining all processes and their instances with all requirements acting on each instance.

**Definition 23: Compliance Measure**

For a set of processes $PS = \{P_1, P_2, ..., P_n\}$ and a set of compliance requirements $RS$, the compliance measure of $PS$ with respect to $RS$, denoted by $CM_{RS}(PS)$, is given by

$$CM_{RS}(PS) = C_{RS}^\Sigma (P_1) \times C_{RS}^\Sigma (P_2) \times ... \times C_{RS}^\Sigma (P_n)$$

For example, taking both process definitions from Example 4, the requirements and instances of execution:

**Example 5: (Compliance Measure Example)**

Degree of Compliance for $P_1$ “Creation of a new user on the DBMS” = $R$(A valid username should be chosen) $\times$ $R$(A ‘good’ password must be chosen) $\times$ $R$(The user and password must be saved)

In this example, each process is prefixed with $P_i.I_n$ where $P_i$ is the process identifier and $I_n$ is the instance identifier.

Transaction Log for creation of a new user on DBMS ($P_1$)

| $P_1.I_1$ | New username: John, New password: secret, (Saved) |
| $P_1.I_2$ | New username: Andrew, New password: password, (Saved) |
| $P_1.I_3$ | New username: John, New password: fido, (Not saved) |

Degree of Compliance for $P_2$ “lodgment of quarterly activity report” = $R$(an activity report must be made) $\times$ $R$(A quarterly activity report must be printed and mailed to the regulatory body) $\times$ $R$(complete in reasonable time).
6.3. Measuring Degrees of Compliance

Transaction Log for lodgment of quarterly activity report (P2)

P2.I3: New report written on the 01-06-2008, Not printed, Unsatisfactory time

Here the Compliance Measure becomes

\[ CM_{RS}(PS) = ((P_1.I_1 \times P_1.I_2 \times P_1.I_3) \times (P_2.I_1 \times P_2.I_2 \times P_2.I_3)) \]

If we use a rating system of good and bad from before then we find that \( CM_{RS}(PS) = \bot \) as there are errors in P1.I3 and P2.I3.

To this point we have worked with both crisp true or false Boolean requirements (in example 3) and fuzzy imprecise requirements (in example 2). In the next section we will provide a method for determining values for imprecise compliance requirements using decision lattices [31,53,103].

6.3.2 Identifying Imprecise Compliance Requirements

When we obtain various departmental policies, such as “Secure the DBMS from well-known attacks” and “Ensure that financial reports are signed in triplicate”, we are faced with the problem of providing consistent values for imprecise compliance requirements.

There is existing work that seeks to monetize “prescriptive policies” [99], i.e., attach monetary penalties for non-compliance. In principle, this could be extended to deal with our problem, by associating differential monetary penalties to varying degrees of non-compliance. Unfortunately monetary penalties alone may not be sufficient to completely describe a policy that has been defined to meet objectives such as “Increase customer satisfaction” or “Reduce environmental impact”. We suggest that a monetary valuation system can be tricky to negotiate and estimates of projected growth of trading can be manufactured to sway audit systems. This is problematic in an organisational setting as an assessment of compliance must be considered across many departments. This problem has been addressed in [223] briefly, with no immediate solution given to the aforementioned monetary valuations.

Our framework is general enough to aid in providing an abstract valuation for the completion of activities that may be combined to produce a degree of compliance across a collection of requirements and processes.
A non-crisp compliance requirement is a process requirement that has multiple acceptance criteria that are difficult to evaluate with a monetary value. As an example, in the process of ‘processing a form’ with a non-crisp requirement that the ‘form must be processed within a reasonable amount of time’, the value associated to this process is not precise and may have varying degree’s of satisfaction. Some interpretations of reasonable amount of time may be ‘2 days’, ‘4 weeks’, or ‘30 minutes’. It is due to the fuzzy nature of imprecise requirement results that these types of compliance requirements may have different values depending on the instance of completion [99].

### 6.3.3 Engineering Compliance Requirements

In this section we present a methodology for acquiring, maintaining and using potentially vague and imprecise requirements.

For imprecise requirements such as “quarterly activity statements must be filed within a reasonable time frame” where reasonable has not been explicitly described, consider that there are a number of possible values that show the processes degree of compliance.

#### Example 6: (Lodgement of quarterly activity statement)

- \(R(‘Activity\ report\ is\ lodged\ in\ 24\ hours’)\) \(\rightarrow\) \(\langle\text{High}\rangle\)
- \(R(‘Activity\ report\ is\ lodged\ in\ 10\ weeks’)\) \(\rightarrow\) \(\langle\text{Medium}\rangle\)
- \(R(‘Activity\ report\ is\ lodged\ in\ 9\ weeks’)\) \(\rightarrow\) \(\langle\text{Normal}\rangle\)
- \(R(‘Activity\ report\ is\ not\ lodged)\) \(\rightarrow\) \(\langle\text{Low}\rangle\)

The statements above assert that if a activity report is filed in 24 hours, the process of lodging the activity has a High degree of compliance. If an activity is not lodged then it’s degree of compliance is Low.

An analyst writing compliance requirements gives examples of possible activity executions with associated values. During process instance execution a human based comparison can be made on the current activity in comparison to the examples provided. Using the previous example requirements, if we find that “the quarterly activity statement is actually filed in 9 Weeks ” then an appropriate value between \(\langle\text{High}\rangle\) and \(\langle\text{Medium}\rangle\) can be given (method for building a degree of compliance description is described in section 6.3.4). The more examples given during the design of processes that associate execution degrees of compliance, the more accurate our values and measures will be at run time. Note that each degree of compliance may have associated with it an n-ary value.
Example 7: (Associating Degree of Compliance to processes)

For the process “User creation” the process requirements are:

- “Selection of a valid username” which has the requirement that a valid username is a username that does not already exist on the computer system.
  
  - If a username exists on the system then a new username should be selected e.g. ‘John’ exists so the next new username maybe ‘John.Smith’, which has a degree of compliance of \( 1 \) and is more preferred over non-completion of the process activity.
  
  - If a username exists and the new user is not created then the rest of the process can not be completed. This is a degree of compliance of \( 0 \).

- “Selection of a ‘good’ password” at this point ‘good’ is not a complete requirement, but the selection of a password is. This is a combination of crisp and non-crisp rules.
  
  - A good password should stand up against possible dictionary attacks, it should not be the same as the users name and should not be easily guessable. A password of mixed case alphanumeric characters (e.g. a-z A-Z 0-9) should be used and the length should be greater than 8 characters e.g. pR0z@c99. If a password meets these standards then the degree of compliance for this activity is \( 100\% \)
  
  - If a password is not at least 8 characters long but contains mixed case alphanumeric characters (e.g. f1D0) then the degree of compliance of this activity is \( 50\% \)
  
  - If a password is 8 characters but does not contain mixed characters (e.g. passwords) then it is considered \( 10\% \)
  
  - If a password is not selected then the degree of compliance for this activity is \( 0\% \)

- “Update the computer system with details of new user”. This activity is undertaken to save the user to the computer system and the requirement is that it must be completed to complete the process. On completion the degree of compliance is \( 1 \), if there is an error then the degree of compliance is \( 0 \).

Implementing crisp rules with an added imprecise degree of compliance gives a method for checking the degree of compliance of various processes. Each degree of compliance can be formed to meet the specification of a specific company or industry. An example of QoS based metrics has been presented in [97,107].
6.3.4 Methodology

To rank and show the degree of compliance we have chosen to use a lattice [53]. The use of a lattice provides a formal setting to represent concept hierarchies [31,103] and in this article, the degree of compliance.

A partially ordered set where every subset of two elements has a lub and a glb is a lattice. Every instance of the c-semiring has been shown to have a lub and glb in [20].

Definition 24: Lattice Definition [19]

A Lattice is a partially ordered set $P$, any two of whose elements have a glb, and a lub.

Returning to our lodgment of quarterly activity report example (example 6). We say that $R$ produces the degree of compliance $R_p$ for a process $P$. During the determination of the degree of compliance a relation of order and equality has been defined such that for the pair \{High, Medium\} there exists a partial order $\leq$ between elements (i.e. High $\leq$ Medium is used to declare High is better than Medium).

For each degree of compliance $R_p$ there exists a partial order where, the partial order is a binary relation represented $\leq$ over the set $R_p$. The partial order over $R_p$ can be used to create chains of comparable degrees of compliance. A chain of compliance degree’s in the set $R_p$ is denoted $RC^i_p$. A chain of compliance is a totally ordered subset of $R_p$.

Definition 25: Lattice Chain

A lattice chain is a totally ordered subset from the overall set of values.

Each lattice chain has the same properties of a partially order set with the addition of totality. Totality is denoted with the $<$ operator to represent that if $a < b$ then $a$ is strictly better than $b$.

Given $x$ and $y$ in a partially ordered set, either $x < y$ or $y < x$.

A partially ordered which satisfies the above is said to be “totally” ordered and is called a chain.

$$a \leq b \text{ or } b \leq a(\text{totality}).$$

Such that.

If $RC^i_p \subseteq R_p$ and $\forall a, b \in RC^i_p, a < b \text{ or } b < a$; then $RC^i_p$ is a chain in $R_p$.
6.3. Measuring Degrees of Compliance

For each element in the example (example 6) we have defined a partial order between degrees of compliance as follows: \{High ≤ Medium\}, \{High ≤ Normal\}, \{Normal ≤ Low\}, \{Medium ≤ Low\}. These orders are shown in the lattices in Figure. 6.3. Figure 6.3(a) shows the ordering of values High and Low, in this ordering High is better than Low.

In Figure. 6.3(b) a non-comparable ordering of Medium and Normal is shown. Figure. 6.3(c) shows the complete partial ordering of all degrees of compliance.

Two chains (see definition 25) exist within the lattice. In Figure. 6.3(c) we show these two chains. The first chain is \{‘Low’, ‘Medium’, ‘High’\} and the second chain is \{‘Low’, ‘Normal’, ‘High’\}.

These non-comparable lattices are common in compliance systems where the preference for elements in the chains is undefined. Elements in the chains can be compared using the c-semiring structure described in the next section.

When creating a new compliance system each new element can be added to the degree of compliance one by one as in [31,103]. For small degree of compliance systems this is a straightforward task. For larger preference systems with large quantities of degrees of alignment we have provided a method for introducing new degrees of compliance values.

**Step 1** Identify each lattice chain. As the degree of compliance structure is built a lattice will be generated.

**Step 2** For each lattice chain identify if there exists values \(v_1, v_2\) with a defined order of \(v_1 < v_2\). Find a mapping of these values to process instances \(p_1, p_2 \in P\) within the system being modeled, such that \(R(p_1) = v_1\) and \(R(p_2) = v_2\). A new instance \(p\) of \(P\) is between \(p_1, p_2\) if \(p_1 \preceq_c p \preceq_c p_2\) then \(R(p)\) can be any value \(v_p\) s.t. \(v_1 \leq v_p \leq v_2\). Where \(\preceq_c\) is the process compliance ordering. A process compliance ordering is to be...
determined by the systems analyst during system modeling. This is the ordering of process completion degree for alternative instance execution evaluation (discussed in Section 6.3.3).

For example, if \(p_1 \Rightarrow \text{Statement is filed in 24 hours}' and \(p_2 \Rightarrow \text{Statement is not filed}'. When introducing a new instance \(p \Rightarrow \text{Statement is filed in 10 weeks}' the value \(v_p) associated to \(p must be between \(v_1 \Rightarrow \langle \text{High} \rangle and \(v_2 \Rightarrow \langle \text{Low} \rangle. When we introduce the new value 'Normal' it stands that \('\text{High}' \leq '\text{Normal}' \leq '\text{Low}'\.

**Step 3** If no such lattice chain exist, then we identify, for each lattice chain with at least one assigned value, either:

- The greatest value \(v_i \) s.t. \(R(p_i) = v_i \) and \(p_i \preceq_c p\)
- The least value \(v_j \) s.t. \(R(p_j) = v_j \) and \(p \preceq_c p_j\).

For example, if \(p_1 \Rightarrow \text{Statement is filed in 24 hours}' and \(p_2 \Rightarrow \text{Statement is filed in 10 weeks}'\). When introducing a new instance \(p \Rightarrow \text{Statement is filed in 9 weeks}'\, a determination that \(p_1\) and \(p_2\) are non comparable as they are saying the same thing. By determining \(p\) is not in a chain with \(p_2\) we say the element \(p \notin RC_i p_2\) if \(p_2 \in RC_i p\, and if \(p \notin p_1, we create a new lattice chain RC_i p\ consisting of the elements \(p_1 \Rightarrow v_1 \leq p \Rightarrow v\). This method can be used to devise degrees of compliance for completing each process instance. Multiple degree of compliance tables may be used to represent varying business goals. We now provide an example of the aforementioned method used to identify the degree of compliance for the process of “user creation” (shown in Figure 6.1).

**Example 8: (Example of Assigning Degree of Compliance)**

For the following example we will refer to Figure 6.1, this is a business process model of the user creation process. The process begins with an administrator requesting a username and password from the user. The user then returns a username and password selection. The administrator logs into the DBMS and runs the create user function - CreateUserCmd(). A username and password are supplied as input and then the DBMS is saved. The requirements for this process are listed in example 7 and example 6. Step one of the methodology is to identify the lattice chains of degree of compliance. For this we consider the method described in section 6.3.4 with the possible execution instances degrees of compliance shown in example 7 and example 6. The username can be either \(\langle 1 \rangle\) or \(\langle 0 \rangle\) and the password either \(\langle 100\% \rangle), \langle 50\% \rangle, \langle 10\% \rangle, \) or \(\langle 0\% \rangle\). Starting with a combination of \(\langle 1,100\% \rangle\) to represent a valid username and a password greater than 8 mixed characters that is not the same as the username or a word in the dictionary. We continue to add the values \(\langle 1,50\% \rangle, \langle 1,10\% \rangle, \langle \text{good},0\% \rangle as a
worse degree of compliance to the current degree of compliance. We then introduce \( \langle 0,100\% \rangle \) to the lattice. This value is worse than \( \langle 0,100\% \rangle \) but non-comparable to \( \langle 1,50\% \rangle \). A new chain for \( \langle 0,100\% \rangle \) is defined as worse than \( \langle 1,100\% \rangle \) and is non-comparable to \( \langle 1,50\% \rangle \). The ordering is dependent on organisational policy as each organisation may value security over non-completion of duties.

### 6.4 Prioritizing repair of non-compliant processes

We use two notions of compliance-driven process repair, these are design repair and execution repair.

In the measurement of degree of compliance we investigate the possibility of analyzing each measure, and its variance from other similar processes. For a single process instance, it is an easy thing for an organisation or departmental unit to follow policy and perform at an optimal compliance level when auditors are watching. For auditing run time compliance there is a need to assess the overall level of compliance even in instances where a process is non-compliant with the requirement policies.

If a process is non-compliant and the number of instances that perform poorly out weigh the number of instances that perform well then it can be assumed that a policy requirement may need to be amended or the activity definitions may need alteration. This idea is called design repair.

**Example 9: (Design Repair of Inconsistent Processes)**

If we added two requirements that said “all new users must be created on the system based on their details of the company HR report” and “the company HR report is to be created based on the details stored on current employees from the computer system user database” to the running example in section 6.2.1, an execution of the process would produce errors as new users could not be created. We would find the instance measure would look like:

\[
\text{Degree of Compliance} = R(\text{all new users must be added based on their details in the company HR report}) \times R(\text{the company HR report is to be created based on the details stored on current employees from the computer system user database}).
\]

Producing combinations where \( R(\text{the company HR report is to be created based on the details stored on current employees from the computer system user database}) \) and \( R(\text{all new users must be added based on their details of the company HR report}) \) will not align and a bad degree of compliance would always be associated with this process.

On the other side of the scale, we review example 3 where of three process instances, two process were performed correctly and one was not completed. The result
of the instance measure in the example was that the combination was a bad process. The process requirements are consistent with each other and it is possible to complete the process; however there is a problem in the activity instances. It is intuitive that if a process is consistently designed then in order to improve performance of the process execution, incentives could be given to complete activities at a higher performance rate.

There are also times within industry where some level of failure is acceptable and further work in defining statistical analysis measures should be undertaken as in [119] to interpret outlier activities and determine a solid variance acceptable for each industry.

Once a business has defined its operational domain and a list of potential activities, there should be a level of consistency in actual completion of processes provided the business logic is correct. There is a still a need for a compliance officer to manage the repair of compliance systems once a breach has been detected.

6.5 Summary and Conclusions

In this work we have presented a method for determining a degree of compliance for business processes. A framework has been provided that can be used in the formation of a general compliance system. We have provided a method for identifying both crisp and imprecise compliance requirements and applying degree of compliance for process instance compliance evaluation. When implemented as a monitoring device on existing compliance frameworks our system can be used to provide transactional monitoring and show valuations of costing and benefits associated to each process.

If an organisation were to adopt these rigorous standards for auditing and preparing the granular run-time transaction statements as we have shown, we would expect that there would be alleviation on further unpredictable behaviour within the organisation. The degree of compliance for processes could also be utilized to identify and repair processes that exhibit non-compliance behaviours.

In this work we have used the term Boolean requirements to describe a set of functional goals that need to be addressed in the execution of general business processes. The use of this term can be interchangeably used to describe functional strategic goals. For imprecise compliance requirements, there is a direct correspondence to optimization objectives used in the description of organisational strategies.
Chapter 7

Toolkit Development

I maintain also that substances, whether material or immaterial, cannot be conceived in their bare essence without any activity, activity being of the essence of substance in general...

GOTTFRIED LEIBNIZ

As part of this thesis, three tools were developed: TextSeer, ServAlign, and Crisis Inducer. In this chapter we will present an overview of the general base level functions implemented in TextSeer which is a packaged set of algorithms used to conduct automated process alignment. In particular, we present methods that demonstrate graph encoding of process models and then also to create decision free task sequences, logic based functions and the accumulation procedure used throughout most of this thesis. The tool itself is a set of library functions, in the degree of approximately 10k lines of java code. An outline of each of the other tools is provided below.

ServAlign was the very first tool developed as part of this thesis. The purpose of ServAlign was to provide a dashboard that showed correlations between strategies and services. The tool was developed as part of a summer project and contained no formal reasoning system. As a diagrammatic widget the tool allowed users to describe strategies and then place them on screen in a diagram. The tool also allowed the user to define services and then to place them on screen in a diagram. The tool contained a wizard that would assist users in the hand correlation of the services that they had designed with the strategies they had designed. ServAlign was developed as an eclipse application and heavily leveraged the diagramming widget provided in GEF. The features of ServAlign included a service search Wizard that was used whenever a user wished to correlate a service to a strategy. The benefit of this tool was
as a first cut proof of concept that was used with industry partners to ensure that the creation of a aligned model would provide value to a business. Through trials with Infosys the tool successfully demonstrated that there was value in correlating services to strategies and provided insights that informed the development of a methodology for algorithmic alignment used in TextSeer.

TextSeer bought together the knowledge obtained through the development of ServAlign and is the general algorithm used in ProcessSeer. The tool used strategies defined in ServAlign and process models defined using the BPMN 2.0 standard with semantic annotations and then provides algorithms to show alignment between the processes and the strategies. TextSeer is more a collection of libraries than a standalone application and provides functionality beyond simple alignment. TextSeer includes algorithms for process similarity measurement, process effect accumulation, process QOS accumulation, effect induction, default reasoning on effects, effect/goal realisation assessment. The purpose of developing textSeer was to move algorithms into a single location in such a way that they could be reused in other applications. The intention is to have the textSeer algorithms used in both ProcessSeer and ServAlign. TextSeer benefits strategic alignment as it provides the underlying algorithms to automatically compute process similarity for integration work, process consistency measurement for compliance work, process to goal realisation assessment, and effect accumulation.

Crisis inducer is a web front end to the default logic reasoner developed as part of textSeer. As default logic is not a conventional logic it was found that users had difficulty describing rules and facts for use in revision style effect descriptions. A standalone tool was created to allow users to test their facts and rules to see the results in real time without the need for running a complete process accumulation task.

TextSeer is built on top of a number of software libraries, including jGraphT, Jung, Orbital, Yaoqiang-BPMN, and jBPT.

jGraphT \[^{1}\] is a java graph library. TextSeer uses its graph object definitions and algorithms. In early iterations of TextSeer, jGraphT was used to compute graph cycles.

Jung \[^{2}\] is another java graph library that in the case of TextSeer is used mainly for visualisation.

Orbital \[^{3}\] is a java math library developed by André Platzer et. al \[^{163}\]. TextSeer leverages the logic components of Orbital for computation of effects.

[^1]: http://jgrapht.org/
[^2]: http://jung.sourceforge.net/
[^3]: http://symbolaris.com/orbital/
7.1 Graph Encoding a Process Model

In the space of BPM, one of the most common issues that is regularly addressed, but not agreed upon is the graph encoding of BPMN models. Typically, most researchers will use a either a Petri-net model encoding or in other cases will invent a graph encoding to meet the purposes of their work in the area. The most common 'sim-

Yaoqiang-BPMN[^4] is a graphical BPMN editor. TextSeer leverages the file parsing system developed in this tool for processing business process models.

jBPT[^5] is a library of code used for process model analysis. Used originally for parsing EPC JSON files from various repositories. This particular library is not heavily used in TextSeer due to a number of technical errors.

Although we have based the tool on ProcessSeer, the accumulation engine is more extensible. In ProcessSeer accumulation is done in a single mode of belief update. This is due to the algorithm design and the accumulation function. In text seeer we have implemented the same base accumulation algorithm with the added benefit of deductive closure at each pairwise accumulation step; additionally, a default logic reasoner has also been implemented such that the accumulation function could be replaced with a default logic reasoner to provide belief revision style accumulation during the process effect accumulation.

Depending on the application of accumulation either a belief update or a belief revision function can be used, in future work we aim to investigate the benefits of each style of accumulation for differing domains. For example, in a customer domain a belief revision function may be more effective as the process may wish to retain knowledge of previous interactions and update its knowledge base accordingly. In the domain of a traffic control process system belief update may be better suited as updates force the removal of bad data.

To test the effectiveness of our tool a randomised process algorithm was also developed. The algorithm produces process models and effects and then attempts to accumulate the effects. This is done to assess the computational time that would be required to conduct alignment across a large amount of organisational data. As expected when the number of processes and effects scenarios increase the computational time needed to compute and effects scenarios grows exponentially.

[^5]: [https://code.google.com/p/jbpt/](https://code.google.com/p/jbpt/)
ple’ (in the case of working on a particular problem that doesn’t need the full BPMN notation) encoding for process models, is based on the WF-net (workflow network) (van der Aalst, 1998); however, to work at the most detailed level it would be advisable to consider the work done by Pieter van Gorp, Remco Dijkman, Marlon Dumas and Chun Ouyang (van Gorp and Dijkman, 2011; Dijkman et al., 2007) in their comprehensive BPMN graph encoding including the execution semantic of a large set of BPMN elements.

In TextSeer, we have developed a graph encoding of BPMN, that extends the jGraphT DefaultDirectedGraph. In addition to this, given that in a process model, each activity may be generalised process, we have extended the Graph class for nodes and then again for Traces. Below in Figure 7.1 we show the a class diagram of the TextSeer graphs.

The graph class is relatively basic and mainly holds references for all of it’s child elements. The primary methods are:

```java
1 . . .
2 1 // Add a node
```
7.1. Graph Encoding a Process Model

```java
public boolean addV(Vertex myV);
// Remove a given node
public boolean removeV(Vertex myV);
// Add an edge
public boolean addE(Edge myE);
// Remove an edge
public boolean removeE(Edge myE);
// Add a pool or subprocess.
public boolean addP(Graph<v, e> pool);
```

... 

The graph edges are much the same. These elements have a pair of associated nodes and are stored as attributes in the graph. The primary edge methods are general getters and setters for other various attributes. Things become much more interesting as graph vertices are defined.

Each graph vertex, stores information on the BPMN element that is within it. Given a typical business process model, each vertex in TextSeer will distinguish between gateways (XOR, AND, OR) and their flow type (splitting or joining). Given that the Yaoqiang BPMN codec is used when loading BPMN2 models, we’ll list the BPMN elements that can be stored in a Vertex based on their Yaoqing element type.

```
/*
BPMN 2 Elements stored in a TextSeer Vertex
*/
** ExclusiveGateway
** ParallelGateway
** InclusiveGateway
** Task
** ServiceTask
** UserTask
** CallActivity
** SubProcess
** StartEvent
** IntermediateThrowEvent
** IntermediateCatchEvent
```
In addition to this we also store semantic effects within the Vertex. In the subsequent section (§7.2) we will describe these in more detail.

To load process models and to perform various operations we have included a set of utility functions, that can be found in the be.fnord.util.processModel.util package. These utilities include model loading, well formness checking and transformation.

be.fnord.util.processModel.util.GraphLoader, uses the method loadModel to parse a file and to create a graph encoding of the process stored in the file. An example of this loading is shown in BPMN2ModelLoadingExample.

```java
// examples/BPMN2ModelLoadingExample.java

/* Load process model */
GraphLoader gLoader = new GraphLoader();
Graph<Vertex, Edge> g1 = gLoader.loadModel("models/Model1.bpmn20.xml");

<<Check 4>>
<<Transform 9>>
```

Once a graph has been loaded, it is required to be cleaned and transformed before the TextSeer functions can be run on it. The first step is to both check and clean the graph. The functions for this have been created in be.fnord.util.processModel.GraphChecker.java. This particular class is required to be run on all graphs that are loaded, as it will locate and identify corresponding gateways.
GraphChecker g1Checker = new GraphChecker();
System.out.println("G1 Test: "+ g1Checker.CheckGraph(g1));
if (!gc.CheckEventsAndGateways(g1)) a.e.println("Issue checking events and gateways");

---

The check function can be broken down into a series of checks and syntactic fixes. The algorithm will make attempts to repair a poorly constructed graph if possible.

```java
/**
 * Check the graph for good structures, fix some if possible and remove the rest
 * This function will add a substructural start and end node to the process which means that the process will only end up with a single start and a single end.
 * @return True if good structured model, false if badly structured model beyond repair
 */
public boolean CheckGraph(Graph<Vertex, Edge> g) {
    // Fix the gateways and boundaries first
    if (!CheckEventsAndGateways(g)) return false;
    <CollapseStartNodes 6>>
    // Do the same for end nodes
    <<AddMissingGateways 7>>
    // Do the same for end gateways
    <<EnsureStronglyConnected 8>>
    // Check if there are correct gateways
    fixGateways(g); // See if we can fix the gateways first then test them
    if (!testGateways(g)) return false;
    // Set start and end nodes:
    g.getStarts();
    g.getEnds();
    return true;
```
The first repair that the graph checker makes is to collapse multiple start nodes. This occurs when a process modeler has identified and modeled multiple start points for a given process model. Because of the way that we aim to use the processes, we create a new vertex and connect outgoing edges to the existing two start events. We do the same for processes that have multiple end events.

```java
// This includes a parallel gateway for splitting multiple starts
if (startNodes.size() > 1) {
    Vertex newStart = new Vertex("newStart-" + UUID.randomUUID(), GraphLoader.ParallelGateway);
    newStart.isAND = true;
    newStart.isSplit = true;
    newStart.isSubstructural = true;
    newStart.corresponding = null;
    g.addV(newStart);
    g.trueStart = newStart;
    for (String s : startNodes) {
        // Create edge to each node
        Edge newEdge = new Edge(newStart, g.vertexRef.get(s));
        g.addE(newEdge);
    }
} else g.trueStart = g.vertexRef.get(startNodes.get(0));
```

A common problem that we found in process models, is that analysts regularly skip including join/close gateways. This typically makes no difference to the executional semantic of a model when XOR gateways are skipped; however, AND and OR gateway joins are required to understand how a model is run. The algorithm is set to correct and guess correct places to insert missing gateway joins.
7.1. Graph Encoding a Process Model

```java
// Do cleanup (add in missing gateway to start of process)
if (g.trueEnd.type == GraphLoader.ParallelGateway && g.trueEnd.corresponding == null) {
    Vertex newStartGate = new Vertex("newStartGate" + UUID.randomUUID(), GraphLoader.ParallelGateway);
    newStartGate.isAND = true;
    newStartGate.isSplit = true;
    newStartGate.isSubstructural = true;
    newStartGate.setCorresponding(g.trueEnd);
    g.addV(newStartGate);
    // Replace edges from the start node to vertices and recreate from the new start node gateway
    LinkedList<Edge> removeList = new LinkedList<Edge>();
    for (Edge e : g.outgoingEdgesOf(g.trueStart)) {
        Edge newEdge = new Edge(newStartGate, e.getTarget());
        g.addE(newEdge);
        removeList.add(e);
    }
    for (Edge e : removeList) {
        g.removeE(e);
    }
    // Add an edge from the start to the new Gateway
    Edge newEdge = new Edge(g.trueStart, newStartGate);
    g.addE(newEdge);
}
```

Used in: `be/fnord/util/processModel/util/GraphChecker.java` on page 131

Finally, the algorithm, creates a copy of the fixed process model and then conducts a strongly connected test to ensure that the final model satisfies the properties of the process graph described previously.

```
// Create copy of graph and connect the ends to the starts
Graph<Vertex, Edge> copy = g.copyGraph(g);
for (String s : startNodes) {
    ...
```
7.1. Graph Encoding a Process Model

```java
for (String e : endNodes) {
    Edge newEdge = new Edge(g.vertexRef.get(e), g.vertexRef.get(s));
    copy.addE(newEdge);
}

// Check if each node is reachable
StrongConnectivityInspector<Vertex, Edge> sci =
    new StrongConnectivityInspector<Vertex, Edge>(copy);
if (!sci.isStronglyConnected()) return false;
```

7.1.1 Decision Free Task Sequence

If after performing a structural check of the process model (with syntactic fixes added); then, we need to perform an operation to produce a decision free process model.

A decision free task sequence, is a model that has all XOR gateways removed. To achieve this, whenever an XOR split is found in a process model, then the process is forked into multiple instances. See Figure 7.2 for a visual example, of the process being forked.

```java
GraphTransformer gt = new GraphTransformer();
LinkedList<Graph<Vertex, Edge>> _decisionless = gt.makeDecisionFree(g1);
LinkedList<Graph<Vertex, Edge>> decisionless = gt.removeDupesFromDecisionFreeGraphs(_decisionless);
for (Graph<Vertex, Edge> g : decisionless) {
    GraphChecker gcc = new GraphChecker();
    boolean isgood = gcc.CheckGraph(g);
    if (isgood) {
        // Create some traces
        LinkedList<Trace> traces = gt.createTrace(g);
        for (Trace trace : traces)
```
The decision free transformation process can be summarized in the snippet method `makeDecisionFree()` below.

```java
/*
 * Create a set of decision free task sequence from an input model. This will split models to have only one start event etc.
 * @param g Input graph
 */
```
7.1. Graph Encoding a Process Model

```java
* @return LinkedList<PGraph<Vertex, Edge>>
*
public LinkedList<Graph<Vertex, Edge>> makeDecisionFree(Graph<Vertex, Edge> g) {
    LinkedList<Graph<Vertex, Edge>> result = new LinkedList<Graph<Vertex, Edge>>();
    LinkedList<String> startEvents = new LinkedList<String>();
    LinkedList<String> joinGates = new LinkedList<String>();
    LinkedList<String> splitGates = new LinkedList<String>();
    ...
    // XOR FOUND
    // Look for paths from each start event to our gate
    for (String startID : startEvents) {
        Vertex startEvent = g.vertexRef.get(g.vertexIDRef.get(startID));
        for (String gateID : splitGates) {
            Vertex gateway = g.vertexRef.get(g.vertexIDRef.get(gateID));
            FloydWarshallShortestPaths<Vertex, Edge> pather =
                new FloydWarshallShortestPaths<Vertex, Edge>(g);
            gp = pather.getShortestPath(startEvent, gateway);
            // Get vertex before the gate and list after the gate
            Vertex predGate = null;
            for (Edge pe : g.incomingEdgesOf(gateway)) {
                predGate = pe.getSource();
            }
            LinkedList<Vertex> succGates = new LinkedList<Vertex>();
            for (Edge pe : g.outgoingEdgesOf(gateway)) {
                succGates.add(g.getEdgeTarget(pe));
            }
            // Remove the gateway
            for (Vertex successor : succGates) {
                // Now we have a path lets build a fragment
                Graph<Vertex, Edge> pg = new Graph<Vertex, Edge>(Edge.class);
            }
        }
    }
```
The algorithm described above will find the first XOR in a graph, and then also its join point. It will then recursively create further decision free graphs between the XOR and the successor as well as all XOR gates after the successor. Once it has a set of traces, it will connect and merge the various start and end points of the traces to form one single set of end-to-end traces.

### 7.2 Effects, WFFs and Accumulation

A business process model (discussed in previous section) represented in the Business Process Modeling Notation (BPMN)\(^6\) is a collection of activities, gateways, events, sequence flows, pools, swim lanes, and message flows. Semantic effect annotations (ADDCITE: HINGE+MORRISON+KOLIADIS) offer a means to reason over business process models. By reasoning with process effects, we are able to capture the organisational operation model, i.e., “what does this process do?” This is important as it

---

\(^6\) see [http://www.bpmn.org](http://www.bpmn.org) for full specifications
allows us to understand what happens as a result of a business process execution; and what execution scenarios a process designer has created for the organisation. In other approaches that rely on syntactical process analysis, no information as to what processes do can be extracted from the process models. This makes pure syntactic analysis difficult when attempting to answer “what” questions about process models.

Previous work in this area (ADDCITE: HINGE+MORRISON+KOLIADIS) has described a method for semantic annotation of business processes. This is an effective way of adding semantic descriptions to process models as it produces reusable artefacts that can be reasoned over. To construct semantically annotated business process models, analysts annotate activities in the model with descriptions of the changes that occur as a result of the activities execution. Such results are referred to as immediate effects of an activity.

In the TextSeer library, effects can be stored in process models and referenced in multiple ways depending on application. These will be covered in various details through the remainder of this section.

Recall our definition of a Process Sequence (see Definition (ADD REFERENCE).), a function $l : N \to \Omega$ maps labels to each Node $n \in N$ in the process. A label $\Omega$ is a tuple containing the node name, type and attached effects.

As BPMN2 has been standardised without a placeholder for semantic effects, we have overwritten the documentation attribute to consist of semantic effects. Below in «models/Model1.bpmn20.xml 11» on page [139] we demonstrate an example of a simple propositional effect scenario attached to the documentation attribute of a task named Task.

```xml
<task completionQuantity="1" id="_7" isForCompensation="false" name="Task" startQuantity="1">
  <documentation id="_7_D_1" textFormat="text/plain">
    <![CDATA[a & b]]>
  </documentation>
  <incoming>_23</incoming>
  <outgoing>_30</outgoing>
</task>
```

At the most basic level, TextSeer will read all text in the documentation attribute
7.2. Effects, WFFs and Accumulation

as an effect scenario and this will be stored as a well-formed formula (a propositional term). Alternatively, we have created a JSON string template that can be used to store richer effects, including details on QoS and instance run time results.

```
<models/Model2.bpmn20.xml 12>

1. . .
2. <userTask activity:assignee="kermit" completionQuantity="1" id="usertask1" implementation="##unspecified" isForCompensation="false" name="Review employee assignments" startQuantity="1">
3.   <documentation id="usertask1_D_1" textFormat="text/plain">
4.     <![CDATA[ _JSONEFFECT {
5.         "Name" : "TestName",
6.         "Type" : "Activity",
7.         "QOS" : {
8.             "COST" : "\$10",
9.             "TIME" : "PT10M",
10.            "SKILL" : "MED",
11.            "UTILITY" : "100"
12.        },
13.         "EFFECT" : [  "A",  
14.             "B",
15.             "C"
16.         ],
17.         "CONSTRAINT" : [  "~A & B & C"
18.         ],
19.         "GOAL" : [],
20.         "RESOURCE" : []
21.     }]
22. ]]]></documentation>
23.   <incoming>flow1</incoming>
24.   <outgoing>flow2</outgoing>
25. </userTask>
26. . .
```

In this string the user can define task labels, types, QoS preferences and measures as well as WFF effects, constraints, goals and resources. In §7.3 we will explain the use of QoS preferences further. Though for now, we'll consider basic semantic effects.
A well-formed formula, is in our work a propositional logic formula. We use propositional variables to represent facts about the world, i.e. \( P = \) It is rainy. A propositional formula is an expression defined over variables that take on the value of true or false. We can build up expressions that describe the world by connecting multiple variables with logical connectives (\( \rightarrow, \land, \lor \)), i.e.

\[
Q = \text{I will catch the bus home}
\]
\[
R = \text{I will walk home}
\]

By using the \( \rightarrow \) inference connective, we can form logical statements:

\( P \rightarrow Q; \neg P \rightarrow R \), either “If it is raining then I will catch the bus home”, alternatively “if it is not raining then I will walk home”.

Given any well-formed formula, an assignment is a mapping of true or false values to each variable in the formula. A satisfying assignment is an assignment of values to the formula such that the formula can be evaluated to true.

When defining the WFF classes, we have leveraged the Orbital reasoner. The general structure of each TextSeer WFF is a formula representation along with a consistency check and boolean sat solver. To compute a satisfying assignment of values in a WFF we call the «be/fnord/util/logic/wff.java/computeAssignments() 13» on page 140 method, which will split a formula into variables and then try assignments of True and False to each to determine if a valid assignment can be made. This method could be replaced by a complete boolean sat solver to improve efficiency in the future.

```
String[] elements = new String[symbols.size() * 2];
int k = 0;
int j = symbols.size();
// Compute all symbols
for (String s : symbols) {
    elements[k] = s;
    elements[k + j] = "~" + s;
    k++;
}
int[] indices;
CombinationGenerator x = new CombinationGenerator(elements.length, symbols.size());
StringBuffer combination;
// For each permutation of the WFF string
```
Given this method for determining satisfying assignments of variables, we can now check for WFF consistency and by taking the union of two WFF’s check consistency of multiple effects. Further in our work we have used the entailment operator. Given a set of premises, or WFF, a conclusion is entailed if every interpretation that satisfies the premise, satisfies the conclusion. This is typically denoted \( \text{PREM} \models \text{CONC} \). As an example, consider the premise “Tweety is murdered”, we may write this as

\[
P = \text{Tweety is dead}
\]

\[
Q = \text{Tweety was killed by someone}
\]

From the premise we can say that a conclusion that is true in every interpretation of the premise, is that “Tweety is dead”. The implementation of entailment is relatively straightforward as Orbital has an entailment inference procedure.
7.2. Effects, WFFs and Accumulation

```java
boolean deduce = false;
this.sigma = logic.scanSignature(s1);
formula = (Formula) logic.createExpression(s1);
this.sigma = logic.scanSignature(s2);
formula2 = (Formula) logic.createExpression(s2);
deduce = logic.inference().infer(new Formula[] {formula},
                      formula2);
return deduce;
```

Finally to round off the logic systems developed in TextSeer, we have also implemented a simply closure method. Given a WFF, it is sometimes necessary to compute all possible inferences that can be made from the WFF. As an example, given the formula: \( A \rightarrow B \land B \rightarrow C \land A \), a transitive closure allows us to conclude that given \( A \), both \( B \) is true, and \( C \) is true. Deductive closure is a little more difficult to define, and in the instance used in this thesis, we have built upon the interpretation of deductive closure described in [171]. For any set of closed wffs \( X \subseteq L \) let \( \Gamma(X) \) be the smallest set in \( Th(X) \) Given some function \( F : 2^L \rightarrow 2^L \) where the function can determine deductive closure and has the following property: \( F(X) \equiv Th(\Gamma(F(X))) \) given the same definition of deductively closed as is used in (R. Reiter, 1980).

The closure computed in the method «be/fnord/util/logic/wff.java/getClosure() 15» on page 142 is an implementation of minimal deductive closure.

```java
Formula formula;
    /// Step one lets get the signature — all the symbols that are used in the wff
...    // See computeAssignmnets() method.
    // Now we have symbols, lets store the symbols and their negation
...    // See computeAssignmnets() method.
while (x.hasMore()) {
    combination = new StringBuffer();
...
    if (eleCount == symbols.size()) {
        /// We now have a sentence that is full of all of our symbols, lets test if it is consistent
        String mSym = ""
        for (String s : _sym)
```
mSym += s + " & "
if (mSym.length() > 1) mSym = mSym.substring(0, mSym.length() - " & ".length());

WFF testForm = new WFF(this.formulaText + " & ( " + mSym + " )");
if (testForm.isConsistent()) {
  boolean part1 = testForm.entails(this);
  boolean part2 = this.entails(testForm);
  if (part1 && part2)
    {
      // Huzzah we have a closure, lets make it CNF
      Logic logic = new ClassicalLogicS();
      if (testForm.getFormula().length() < 1)
        return ""
      formula = (Formula) logic.createExpression(testForm.getFormula());
      Formula result = ClassicalLogicS.Utilities.conjunctiveForm(formula, true);
      DefaultClausalFactory myFacts = new DefaultClausalFactory();
      ClausalSet myClauses = myFacts.asClausalSet(result);
      Formula f = myClauses.toFormula();
      return f.toString();

    } else if (part1)
      {
        if (testForm.getFormula().length() < 1) continue;
        currentBest = testForm.getFormula();
      }
    }
  }
}

// We have found a closure that has less than all variables available to us. Lets reduce.
7.2. Effects, WFFs and Accumulation

// We hack the reduction because the orbital ClausalSet class
doesn't implement remove very well
if (currentBest.length() > 1) {
  Logic logic = new ClassicalLogicS();
  WFF newW = new WFF(currentBest);
  // New Sentence
  formula = (Formula) logic.createExpression(newW.
    getFormula());
  Formula result = ClassicalLogicS.Utilities.
    conjunctiveForm(formula, true);
  DefaultClausalFactory myFacts = new
    DefaultClausalFactory();
  ClausalSet myClauses = myFacts.asClausalSet(result);
  ClausalSet myClauses3 = myFacts.asClausalSet(result);
  // Old sentence
  Formula formula2 = (Formula) logic.createExpression(this.
    getFormula());
  Formula result2 = ClassicalLogicS.Utilities.
    conjunctiveForm(formula2, true);
  DefaultClausalFactory myFacts2 = new
    DefaultClausalFactory();
  ClausalSet myClauses2 = myFacts2.asClausalSet(result2);
  // Remove all repeated elements
  myClauses.removeAll(myClauses2);
  Iterator<Clause> i = myClauses.iterator();
  HashSet<String> myClosures = new HashSet<String>();
  while (i.hasNext()) {
    Clause c = i.next();
    Formula f = myClauses3.toFormula();
    String alt = "~" + c.toArray()[0];
    if ((c.toArray()[0].toString().trim().charAt(0) "+").
      compareTo("~") == 0) alt = c.toArray()[0].toString()
        ().substring(1, c.toArray()[0].toString().length())
    ;
    String newString = removeFromString(f.toString(), "" +
      c.toArray()[0]);
newString = removeFromString(newString, " " + alt);
if (newString.trim().charAt(newString.length() - 1) == '&
    ') newString = newString.trim().substring(0,
    newString.length() - 1);
String s = new WFF(newString).getClosure();

myClosures.add(s);
}
// Get the biggest
ClausalSet biggest = myClauses;
for(String s : myClosures){
    Formula formula4 = (Formula) logic.createExpression(
        new WFF(s).getFormula());
    Formula result4 = ClassicalLogicS.Utilities.
        conjunctiveForm(formula4, true);
    DefaultClausalFactory myFacts4 = new
        DefaultClausalFactory();
    ClausalSet myClauses4 = myFacts4.asClausalSet(
        result4);
    Formula f = myClauses4.toFormula();
    if (!new WFF("( " + f.toString() + " ) & (" + this.
        getFormula() +" )").isConsistent()) continue;
    if (myClauses4.size() > biggest.size() ) biggest =
        myClauses4;
}

// Lets make sure we have all the original stuff
Formula testForm = (Formula) logic.createExpression(this.
    .getFormula());
Formula result9 = ClassicalLogicS.Utilities.
    conjunctiveForm(testForm, true);
DefaultClausalFactory myFacts9 = new
    DefaultClausalFactory();
ClausalSet myClauses9 = myFacts9.asClausalSet(result9);
biggest.addAll(myClauses9);
Formula f = biggest.toFormula();
return f.toString();}
7.2. Effects, WFFs and Accumulation

return this.formulaText;

To compute minimal deductive closure, the tool simulates the symbolic string to create a fixed point WFF of variables that can be inferred. The method is built on the assumption that all facts and knowledge are known; which means that only the variables given to the method are valid in the logical language that reasoning is conducted over.

Given our operations, we now describe the use of accumulation, or belief update. Each WFF annotation can then be accumulated using a function to produce a semantic description of the process model. Let \( e_a \) be a set of effects (or a singleton immediate effect) associated with an activity \( a \) within a process \( P \). Given two sets of effects \( e_i \) and \( e_j \), let a function \( acc(e_i,e_j) \) (defined in [95]) return the accumulation of both immediate effects which is a set of possible effect scenarios. The algorithm in \( \text{pairwise_acc()} \) (below) demonstrates a pairwise accumulation operation, as described in the theory. Given two WFF,

```java
// If the union of the WFF is not satisfiable (i.e. consistent)
if (!source.isSat(target, KB)) {
    // Remove symbols from the source until consistent with target
    if (!source.eval(target, KB)) { // Not sat so compute maxSubsets
        LinkedHashSet<WFF> results = new LinkedHashSet<WFF>();
        LinkedHashSet<String> cleanedResults =
            new LinkedHashSet<String>();
        // Recursively repeat accumulation function with fewer and fewer source nodes
        // until maximally consistent subset is found.
        do {
            Set<WFF> correctDepthResults =
                makeCorrectDepthResults(source);
            for (WFF newResult : correctDepthResults) {
                LinkedHashSet<WFF> returnPairResults;
                returnPairResults =
                    recPairAcc(newResult, target, KB, distro);
                results.addAll(returnPairResults);
```


7.2. Effects, WFFs and Accumulation

```java
    }  
    if (results != null && results.size() > 0) 
        break;  
    currentDepth--;  
    if (currentDepth < 0) 
        break;  
    while (currentDepth > 0) ;  
    ...  
} else {  
    WFF resultingScenario =  
        new WFF(source.getEffect(target.getFormula(), KB ));  
    LinkedHashSet<WFF> returnList = new LinkedHashSet<WFF>();  
    returnList.add(resultingScenario);  
    return returnList;  
}
```

The result of using the above logical functions can be seen in the following examples:

\[(p \rightarrow q), (m \rightarrow p \lor q) \models (m \rightarrow q) \models true\]
\[(p \rightarrow q), (m \rightarrow p \lor q) \cup \neg(m \rightarrow q) \models true\]

Checking if \(\neg(a \land b) \lor \neg c\) is consistent: true
Checking if \(\neg(a \land b) \lor \neg c \models (a \land b) \rightarrow \neg c\): true
Effect scenario resulting from acc: \((\neg(a \land b) \lor \neg c) \land ((a \land b) \rightarrow \neg c)\)
Checking if \((\neg(a \land b) \lor \neg c) \land ((a \land b) \rightarrow \neg c)\) is consistent: true
Effect scenario resulting from acc: \((a \land b)\)
Checking if \((a \land b)\) is consistent: true

In this section we have described the logical framework implemented in this thesis. In the following section effect accumulation is extended to include accumulation of QoS.
7.3 QOS Preferences

Quality of service preferences can be measured using c-semirings. Therefore in TextSeer we have implemented a number of instances of the c-semiring framework.

Definition 26: c-semiring [20]
A c-semiring is a tuple \( \langle A, +, \times, 0, 1 \rangle \) such that:

(i) \( A \) is a set and \( 0, 1 \in A \)

(ii) \( + \) is called the comparison operation. It is commutative (i.e. \( a + b = b + a \)), and associative (i.e. \( a + (b + c) = (a + b) + c \)), and the identity element for \( + \) is \( 0 \) (i.e. \( a + 0 = a \)). \( + \) is idempotent and gives partial ordering \( \leq \) where \( a \leq_s b \) implies \( a + b = b \).

(iii) \( \times \) is called the combination operation, is an associative operation such that \( 1 \) is its unit element and \( 0 \) is its annihilator (i.e. \( a \times 0 = 0 \) and \( a \times 1 = a \)).

(iv) \( \times \) distributes over \( + \) (i.e. \( a \times (b + c) = ab + ac \))

\[ \square \]

A c-semiring has the properties of partial order [20] such that given any c-semiring \( S = \langle A, +, \times, 0, 1 \rangle \), a partial order \( \leq_S \) over \( A \) exists such that \( a \leq_S b \) iff \( a + b = b \).

A partially ordered set (poset) is a grounded set \( X \) with a partial order, induced by an operator \( \leq \). Poset have the following properties:

- \( a \leq a \) (reflexivity);
- if \( a \leq b \) and \( b \leq a \) then \( a = b \) (antisymmetry);
- if \( a \leq b \) and \( b \leq c \) then \( a \leq c \) (transitivity).

Our partial order class must hold with each of the above properties. To define instances of our class we want to pass pairs of values in, where the pairs show ordering. The set of pairs \( \{ \langle A, B \rangle, \langle B, C \rangle \} \) should be the grounded set \( X = \{ A, B, C \} \) with an ordering where \( A \leq B \) and \( B \leq C \) and \( A \leq C \).

Our partially ordered set class satisfies these properties by implementing order operations.
To store our pairs, the best structure to start with is a map. Where each each key represents the first value of a pair and the value represents the second value of a pair. This will cover typical cases where we get inputs of: \( \langle A, B \rangle \) or \( \langle B, C \rangle \); however, when we get multi-values e.g. \( \langle A, B \rangle \) and \( \langle A, D \rangle \) we will need to make sure we don’t write over old map pairs. When new data comes in we’ll need to run it through a data preparation function.
7.3. QOS Preferences

```java
static <T> Pair<T,T> P(T a, T b) {
    return new Pair<T,T>(a,b);
}
```

```java
// When a new list is created, store the data in a map
public <T> Map<T,HashSet<T>> List(T... elements) {
    // To ensure that we don’t write over existing map keys, process the data first.
    return computeOrders((java.util.List<Pair<T, T>>) new LinkedList<T>(Arrays.asList(elements)));
}
```

The above list function allows us to input an n-ary list of generic elements, though looking at line 7, shows that the elements should be generically typed pairs. For example, a use of this function could be:

\[\text{List}(p('A', 'B'), p('B', 'C'))\]

Due to the use of these generic types, whatever datatypes that is used for the pairs should be accepted by the list and poset.

The next stage of the development is relatively straightforward. We are simply going to create a map with a set of values from incoming pairs. So that the first element of a pair is the key and the second element is added to the values set.

```java
public <T> Map<T,HashSet<T>> computeOrders(List<Pair<T, T>> s) {
    orders = new TreeMap<T,HashSet<T>>();
    backwards = new TreeMap<T,HashSet<T>>();
    for (Pair<T,T> p : s) {
        if (orders.containsKey(p.getFirst())){
            ((HashSet<T>)orders.get(p.getFirst())).add((T) p.getSecond());
        } else {
            HashSet<T> h = new HashSet<T>();
            h.add(p.getSecond());
        }
    }
```
Once the map exists a key will have an associated set of values that are worse, i.e., key \( \leq \) value. Though, stopping implementation misses the property of transitivity.

To implement transitivity, a second map is created that will hold the reverse of the order map. That is a key will have an associated list of values that are better than it. We then have an extra function that will loop to ensure transitive closure for all elements of the new map.

```java
SETTINGS: DATASTRUCTURE on page 150 | INCLUDED BLOCKS: 20 on page 151

// Do transitive closure
/* Setup some variables to store hash of the existing maps */
int backHash;
int orderHash;
int _backHash;
int _orderHash;
// Loop through a closure function until all transitive elements are included in backwards map
do{
    backHash = backwards.hashCode();
    orderHash = orders.hashCode();
doClosure();
    _backHash = backwards.hashCode();
    _orderHash = orders.hashCode();
} while (backHash != _backHash || orderHash != _orderHash);
return (Map<T, HashSet<T>>) orders;
}

// Not completely optimal, this function will search all orders and build a transitive set of values.
public <T> void doClosure() {
```
for (T key : ((TreeMap<T, HashSet<T>>) orders).keySet()) {
    for (T ele : ((HashSet<T>) orders.get(key))) {
        if ( ((TreeMap<T, HashSet<T>>) backwards).containsKey(ele)) {
            ((TreeMap<T, HashSet<T>>) backwards).get(ele).add(key);
            if ( ((TreeMap<T, HashSet<T>>) backwards).containsKey(key) ) {
                ((TreeMap<T, HashSet<T>>) backwards).get(ele).addAll(((TreeMap<T, HashSet<T>>) backwards).get(key));
            } else {
                HashSet<T> h = new HashSet<T>();
                h.add(key);
                h.add(ele); // Ensure satisfaction of reflexivity
                ((TreeMap<T, HashSet<T>>) backwards).put(ele, h);
            }
        }
    }
}

At this point our data structure is complete, we can create new poset (without operators) using by first declaring a new instance of poset:

Poset p = new Poset(); and then filling a list of pairs, e.g.,

p.List(P("a", "b"), P("b", "c"), P("a", "d"), P("d", "c"));

Which will result in the following structure in the backward map of values:

{b=[a], c=[d, b, a], d=[a]}

To complete the poset class we will implement three operators, less than (LT), less than or equal to (LEQ), and equal to (EQ).

public <T> boolean leq(T a, T b) {
    if (backwards != null && (backwards.containsKey(a) || backwards.containsKey(b))) {
        if (
Less than or equal to (LEQ) is the first implemented function, and is relatively simple to understand. If the second parameter has a key entry in the reversed map then open the value set and search for the first parameter. If the first parameter is in the value set then it has a relative ordering to the second parameter, either less than (LE) or (EQ).

From here the remaining operators are trivial. Based on the property of antisymmetry: if \( a \leq b \) and \( b \leq a \) then \( a = b \) we can show equivalence with the functional definition of EQ.

The less than method (LE) is relatively straight forward as well. To compute less than (LT), we evaluate true for any pair that has the relation LEQ and that does not have the EQ relation.

With a poset class implemented, we then construct instances of the c-Semiring framework. In particular we focused on COST, SKILL, and TIME. Each QoS preference class, implements a standard structure including initialisation and then also the comparison \( \oplus \) and combination operator \( \otimes \) from the c-semiring definition.
## 7.3. QOS Preferences

```
public class MAX_TIME extends Preferences<Float> implements PREF_FUNC {
    // Set TOP and BOTTOM Values (set from an enum)
    type range = Ranges.type.TIME;
    static String bot = Ranges.LNGTIME;
    static String top = Ranges.SHRTTIME;

    </ComparisonOperator 25>>
    </CombinationOperator 26>>
}
```

Importantly, for each instance, the comparison operator must reflect whether the preference works to maximize or minimize a value.

```
public <T> boolean compare(T aa, T bb) {
    if(aa == null || bb == null) return false;
    if( !aa.getClass().equals(DateTime.class) ){
        return false;
    }
    return (Convert((String)bb)).isBefore(Convert((String)aa));
}
```

The combination operator is very similar across all instances, differing mainly on the use of either addition or multiplication. For set based preferences, the preference poset must define it’s own combination function.

```
public <T> String combine(T aa, T bb) {
    if(aa == null || bb == null) return "";
    if( !aa.getClass().equals(String.class) ){
```
Finally, we have also implemented a generic accumulation function for QoS, in «QOSpairwiseAccumulation 27» on page 155

```java
public Qos pairwise_acc(Qos source, Qos target, JSONEFFECT _src,
                        JSONEFFECT _trg) {
    Qos _result = new Qos();

    // Process the goals to check which objective functions we’re using
    HashSet<String> goals = new HashSet<String>();
    for(String s: _src.GOAL)
        goals.add(s);
    for(String s: _trg.GOAL)
        goals.add(s);

    // Handle Cost Accumulation
    PREF_FUNC costPref = null;
    if(goals.contains("MINPRICE"))
        costPref = new MIN_COST();
    else
        costPref = new MAX_COST();

    // Handle Time Preferences
    PREF_FUNC timePref = null;
    if(goals.contains("MINTIME"))
```
In this section, we’ve shown the implementation of a QoS preference structure and algorithms for accumulating QoS across a process model in a pairwise manner. In this chapter we have described the building blocks of the implemented tool created as part of this thesis. At the time of publishing the tool has been released as a stable version 1.3.1. Included in the tool are the following extra functionality. The tool has been released as an opensource library released under the apache2 license.

Process Similarity - written by Evan Morrison, a process similarity measure using Stanford NLP.

Abductive - written by Evan Morrison, an abductive reasoner, will take in a knowledgebase, some possible actions effects (facts) and an observation. The reasoner will

7The tool can be downloaded from http://www.github.com/edm92/textseer/
then find possible actions that could be performed together.

Accumulation - written by Evan Morrison, accumulation demonstrates a method for conducting belief update across effects annotated to a business process model. The procedure is documented in research papers by Hinge et. al. 2009, Morrison et. al. 2011, 2014

BPMN2 Model Loading - written by Evan Morrison, demonstrates the use of bpmn model loading and graph checking functions.

Decision Free Graph Converter - written by Evan Morrison, documented in work described in Morrison et. al. 2011, 2014. This demonstrates the process for loading a graph and then removing all xor decisions from the graph.

Default Logic - written by Evan Morrison, a default logic reasoner based on Reiters Default logic, this is a lightweight and incomplete implementation of a default logic reasoner. No decision procedure has been documented for this tool and it will only produce results for a subset of default logic problems.

Order Constrained Permutation - written by Evan Morrison, templated functions written to compute permutations that hold combination based orders.

To date the tool has been used to assist in work completed by XU et. al, Hinge et. al, Horesch Khloe et. al, Le et. al. and has been used in the commercial product ServAlign developed by the CRC for Smart Services.
Chapter 8

Conclusion

"Contrariwise," continued Tweedledee,
"if it was so, it might be; and if it
were so, it would be; but as it isn’t,
it ain’t. That’s logic."...

CHARLES DODGSON

In this chapter I will conclude the thesis. The main contribution of this thesis was to describe and formalise a framework that could be used to assess strategic alignment between organisational strategies and business processes. The sub-contributions include a framework for semantic process composition, a framework for process integration, a framework for compliance management, and finally a framework for strategic alignment and realisation. In addition to this, this thesis has an implemented toolkit, and the frameworks presented have been used to help construct the case study listed in the appendix. The proposed set of frameworks are described generally and can be used with a variety of formal representations and organisational modeling notations.

8.1 Summary of contributions

I approached the question of whether a general framework for strategic alignment could be constructed by breaking the concept into a series of sub questions. The first of which was “is it possible to conduct end-to-end enterprise process analysis?” Which I have addressed in §3 through the creation of a composition framework that can be used to analyse processes drawn from the enterprise process repository composed together in all conceivable ways. I have also included a description of how to
8.1. Summary of contributions

conduct analysis on these end-to-end processes through semantic analysis, quality of service analysis, and case analysis. The resulting framework achieves this firstly by demonstrating an encoding of a business process is a graph, and then how to break that graph into a set of traces, and then how to string sets of traces with the traces of other processes from the repository and then finally how to compute annotations and the accumulation of annotations across the traces. The resulting framework is general and can be applied in many process notations, in particular we have focused on BPMN; however, it is conceivable that the framework can be applied to any number of other notations such as EPCs and Petri nets.

The second question that I answered in this thesis was "given a set of process antecedents can a new process be created such that the distance from the new integrated process to each of the antecedents is minimal?" I answered this question through the creation of a process integration framework shown in §7. The process integration framework provides a method for this is analysis between two or more processors at a structural and semantic level. A series of measures were also presented in this framework that could be used to assess semantic difference at the level of effects, quality of service, and case. In addition to this we have provided a method for assessing similarity between composed processes from an enterprise portfolio. Conceivably this means that it is possible to compute the difference in operations at a structural and semantic level between two different organisations.

The third question answered was whether a general framework for strategic alignment existed? I answered this question with the creation of a method for assessing strategic alignment through goal realisation shown in §5. In addition to this I provided a strategic ontology that could be used to help create strategic documents which could then be decomposed into strategic goals, strategic plans, and strategic optimisation objectives. This framework can be used to assess goal realisation, plan realisation and quality based on optimisation objective realisation.

The fourth question answered was how compliance management could be used and assessed in addition to strategic alignment. To answer this question I have constructed a compliance assessment framework, this framework can be used to assess compliance at a design time level and a runtime level shown in §6. The benefit of this assessment framework is that compliance requirements can be described in general or broad language and assessed in a spectral level of degree.
8.2 Benefits and application

For each of the above contributions, there has been an algorithmic implementation that demonstrates an application of the theoretical framework being shown. In addition I have utilised each of the frameworks in the creation of a case study that describes a pension administration organisation. The ramification and benefits of the above contributions are numerous. These include the provision of new measurement frameworks for which organisations can use to assess alignment, compliance, and to conduct analysis over existing process repositories. Given a set of organisational strategies and a process portfolio, it is now possible to describe the relationship for which the processes fulfil the strategies, and the degree to which strategies are being realised by the processes. Given a set of compliance requirements and a process portfolio, it is now possible to describe the degree to which the process design is compliant against the compliance requirements, and it is possible to conduct audits over process logs. Software arising from this work include:

- TextSeer - a general library of process functions for use in third-party applications.
- Crisis inducer - a default logic library, providing functions and methods for computing default extensions and abductive proofs.
- ServAlign - a software product created for the Smart Services CRC. Used to model the correlation of business services to organisational strategy.

8.3 Shortcomings

Although the tools provide an automated method for completing strategic alignment there are a number of shortfalls and potential research directions that can be followed. The algorithms and methods for accumulation of effects and effect scenarios grows at an exponential rate and as such larger organisations are modelled the time taken to complete automated alignment reasoning would also grow exponentially. This has not been considered to be a major shortcoming as it is assumed that a alignment is an activity undertaken primarily during strategy formation. Typically a strategy formation may take many months to achieve and are such executional times greater than one week do not hinder the benefits provided.
Our approach and methodology for conducting this research has suffered through the lack of large-scale open examples for us to test our system and framework on. An appendix has been added of a reasonable sized case study which may be used in future work by other researchers. Additionally we have made a number of assumptions primarily that the realisation method we have for the correlation of processes and strategies will work across all domains. Further investigation and study into business case studies from varying and unique domains will assist in rectifying this.

8.4 Future work

It is the authors intention to continue work in the area to further develop the frameworks described in this thesis, and to provide organisational alignment frameworks that will provide organisational alignment assessments using each of the frameworks presented in this thesis. In addition, the author will continue to work on similarity measures and integration between process models; in particular focused on the application of language used within processes as a further measure in the distance metrics.

Acknowledgements

Work on this project has been supported by the Australian CRC for Smart Services. A greatful thank you for all of their support during the completion of this thesis.
Appendix A

Case Study

To evaluate the frameworks presented in this thesis, a large-scale case study was used. This case study has been provided as during study no other case studies of this size could be found. Typically researchers in the space of requirements engineering either use toy or simple examples. The simple and toy examples used in other articles is irrelevant as other articles demonstrate single components of a larger idea or framework. As this thesis sets to demonstrate alignment and alignment are based on a large set of related components these simple toy examples cannot be used.

The Pension Administration Corporation (PAC) is a leading administrator in Australia, and it currently administers medium to large sized pension funds in both the public and private sectors PAC has the aim of becoming Australia’s number one company in pension services, by supplying high quality administration services to trustees, members, and employers.

The mission of PAC is to

1. Enhance the management and use of pension services,
2. Build capacity and capability within the pension services industry.
3. Develop new sustainable services for future national and personal wealth.

The objectives for PAC are:

1. Institution of a clear, comprehensive and coordinated roadmap to being Australia’ primary pension services company.
2. To sustain leadership in Australian pension services.
3. Move towards an immersive digital experience for their customers.
4. Make current processes, focused around the customer (customer centric).

5. Facilitate new and improved educational services to empower their customers and employees.

6. Win new clients with a clear product proposition at low acquisition price.

7. Demonstrate a sense of social responsibility, and having respect for the interests of the local community.

8. Conduct its operations in compliance with the principles of sustainable development.


The primary driver of PAC is the growth, advancement and conduct of its core business, of providing pension scheme administration and related services. PAC currently provides such services to the trustees of retirement benefit funds. Due to the significant regulatory changes in the pension industry PAC is aiming to conduct business process development project, with the objective of bringing all of its legacy systems and processes up to date. It will do this by implementing new processes that comply with the regulatory obligations.

A.0.1 Market

Currently, PAC provides services to a number of state owned and industry funds. They offer all levels of administration services, including digital customer portals, mail-in form driven systems, and call centres. Competitors in the market PartnerRetire have been implementing a PensionGeneration program which comprises 9 new IT systems. They aspire to provide their commercial enterprise with integrated systems, straight-through processing and advanced analytics.
A.1 Company Overview

A.1.1 Business Opportunities

PAC is engaging in a revolutionary re-engineering process. Due to changes in the pension services environment, PAC has been tasked with reviewing all areas of their core business operations.

As a priority PAC, must move towards a streamlined digital solution offering. The aim of which is to increase the rate at which customers can satisfactorily resolve any issues or requests. PAC is working hard to retire and minimise usage of the mailing unit, which processes physical paperwork. By increasing the ease of digital access and minimising paper systems, PAC allows existing members to continue to utilise the existing service offering while also encouraging younger members to switch to and use the digital product offerings.

By investing in digital solutions PAC wishes to broaden its product range and channel distribution. Over the coming years, the aim is to become industry-leading in online and mobile channels with an emphasis on friendly and efficient customer interaction. PAC’s priority action for business development is increased communication with stakeholders and customers to promote the importance value and benefits of pension services now and in the future. The purpose is to work with providers of the new pension services and to acquire new technologies that will improve their service offerings. They have a commitment to support research, innovation and skills development and leverage the country’s innovation to create new services for the future.

PAC aims to enhance and develop its product range by incorporating the latest in the analytics management and monitoring. Through the use of data, the aim is to deepen customer relationships, learning more about how best to engage and offer tailored solutions for members.

PAC wishes to address operational challenges in providing a long-term employee retention strategy. This includes investment in a new training platform which has been offered to all existing employees as well as working closely to build a long term career plans the future engagement with our best staff.
A.1.2 Business Development

PAC’s research into the current state of its operations has identified the following areas for PAC to focus most of its attention on during its change process:

1. Core processes are repeated across multiple departments
   (a) Where possible integrate processes, merging similar processes into single processes.
   (b) Identify straight through processes.

2. Changes in laws have made various processes non-compliant.
   (a) Maintain compliance with existing legislation and updated to meet new legislative requirements.
   (b) Align all processes to the strategic plan.

A.1.3 Position

The aim of rolling out new systems is to ensure that PAC can compete over the long term. There is a big push towards developing a digital engagement suite.

<table>
<thead>
<tr>
<th>Organisation Key Industry:</th>
<th>Pension Services</th>
</tr>
</thead>
</table>
| Identification of organisational niche and key services | Pension administration  
Mailing centre  
Call centre  
Member management  
Newsletter  
Digital Portal |
| Reason for existence: | Providing administration services for pension funds that focus primarily on investment and returns for members. |
### A.1. Company Overview

**How is the organisation unique:**
- All call centre operations are run within Australia
- Rich history providing administration to Australian.

**Key customers and users**
- OLGST
- SECOND
- PEFS
- PenServices
- PTAST
- TPN

**Five most important strategies being pursued currently:**
- Increased growth
- Winning new customers
- Producing new products
- Increasing productivity
- Retaining high quality products and standardising core systems.
- Improving communication time.
- Increasing value from existing customers

**Major political and management issues are currently being faced:**
- Major changes to pension legislation.
- Key goal is to maintain compliance with new rolling legislation.
### Name: Ensure members contributions are allocated

#### Functional Goal

Aim: to improve the automation for new member contributions through an automation process.

Current State: Non-process centric  
Desired State: Process-centric

#### Purpose

Ramification if complete

Description: When a contribution is made by an employer or member, the payment will be made automatically.  
*This will increase value to the organisation from existing customers*

#### Stakeholders

Internal: Processing staff, accounting  
External: Fund members, Employers
<table>
<thead>
<tr>
<th>Strategy</th>
<th><strong>Name: Member details are kept up to date</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Functional Goal</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Aim: to improve the automation for existing member change of details.</td>
</tr>
<tr>
<td><strong>Current State:</strong></td>
<td>There exists a manual process for this activity.</td>
</tr>
<tr>
<td><strong>Desired State:</strong></td>
<td>Wish to provide digital form access and automation.</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Ramification if complete</strong></td>
<td>Description: Member details will be kept up to date. <em>This will improve communication time.</em></td>
</tr>
<tr>
<td><strong>Stakeholders</strong></td>
<td>Internal: Processing staff</td>
</tr>
<tr>
<td></td>
<td>External: Fund members</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strategy</th>
<th><strong>Name: New digital core consolidation process created</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Functional Goal</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Aim: Standardise the consolidation process, by first incorporating activities into one process in the new process engine. Then in a future project automate and digitise the system.</td>
</tr>
<tr>
<td><strong>Current State:</strong></td>
<td>Dispersed processes for consolidation.</td>
</tr>
<tr>
<td><strong>Desired State:</strong></td>
<td>Centralised process for consolidation.</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Ramification if complete</strong></td>
<td>Description: Operations will become more standardised, and PAC can begin to <em>offer new products and services</em> associate to the new standardised process.</td>
</tr>
<tr>
<td><strong>Stakeholders</strong></td>
<td>Internal: Processing staff</td>
</tr>
<tr>
<td></td>
<td>External: Fund members, competitors</td>
</tr>
</tbody>
</table>
A.1.4 Outcomes

The outcome of the re-engineering process is to position PAC as an industry leader and ensure the future sustainability of the company. Key to the success of this goal is the delivery of the following:

1. A core set of reference models.
2. Standardised forms.
3. Tight integration between digital platform and current systems.
4. All processes conducted at PAC must be compliant with new laws.

A.2 Stakeholders

**Customers**: Improved communication and touch points. It will be easier to communicate with PAC on the completion of this project. In addition, the customers will receive more improved information about their current accounts and will be better able to plan for their retirement.

**Processing staff**: Currently have to deal with customer complaints and lack of responsiveness. The new systems should reduce the complexity for the processing team. Overall the system should be much easier to learn and to keep up to date with. There should be a reduction in case processing times, which should lead to give staff better opportunities to personalise the service to customers.

**Regulatory bodies**: Required PAC to be compliance and conformant with all relevant legislation.

**Trustee**: Trustees is the fund board. They make decisions on who will administer their fund and where to invest.

<table>
<thead>
<tr>
<th>Actor</th>
<th>Description</th>
<th>Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## A.2. Stakeholders

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Members</strong></td>
<td>are fund clients of a pension Fund; each Fund is managed by the Fund Trustee.</td>
<td></td>
</tr>
<tr>
<td>A fund client will have an account with typically one fund; however, they may have multiple funds.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Fund clients</strong></td>
<td>will interact with their administrator (in this instance PAC), as the administrator stores all of their information and details of how they wish their pension to be invested.</td>
<td></td>
</tr>
<tr>
<td>Fund clients may update their details, with a paper form, it is hoped that by rolling out a new system that fund clients will also benefit from easier interaction to the administrator with the digital form system.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fund clients are external entities.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-2</td>
<td><strong>Trustee</strong></td>
<td></td>
</tr>
</tbody>
</table>
### A.2. Stakeholders

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
| **A-3** | *Regulator*  
The pension regulator ensures that the trustee is doing the right thing with the investment monies it also ensures that all laws and regulations are followed, and all reporting is done and complete accurately. | Maximise legislation and regulations that are enforced |
| **A-4** | *Processor*  
Processing staff members are the team members working for the administrator. Their job is to complete the processing tasks assigned to them by the PensionProcessSystem. For example, if a customer rings up to change their details with the pension administrator, call center staff will direct the call to the appropriate processing staff member who will then handle the case and address the customer request. |   |
A.2. Stakeholders

Call Centre Staff

Call Centre staff are the workers on the ground that answer Fund clients when they call to request information about their accounts or to update their details.

Typically a major issue with running a call centre is that there is high employee attrition, minimising stress on these team members is important. Ensuring minimal downtime and minimal retraining is also important. It is also due to the high attrition rates that PAC wishes to implement a New Process Engine as the Old Process Engine training times are moderately high.

Reduce complexity of New Process Engine.
Minimise staff attrition by minimising stress on staff and minimising system downtime.
Have an easy to use New Process Engine.
Make Call Centre Staff happy by minimising Fund client’s complaints.
### A.2. Stakeholders

#### A-7 Mailing Unit

The **mailing unit** at the pension provider is the unit that receives all **member** mail requests. These units have evolved at last 10 years. Previously they would have been a big mail centre where people would open the letters by hand. Staff would read the letters manually and then direct the letter to the correct department. Now the system is automated with optical character recognition and the requests are directed to the processing staff **members** automatically. Over time, the aim of **PAC** is to retire the **mailing unit** and handle all forms digitally with the **digital form system**.

#### A-9 PensionProcessSystem

**New Process Engine**

#### A-11 MyCasePictures

**MyCasePictures** is a case tools program. When a letter is received in the **mailing unit**, it is opened and scanned into the case tool. A case also includes notes. A case is a specific interaction between the fund client and the fund. For example, if a member wishes to change their details they will submit a change member’s detail from. The **mailing unit** will open and scan the letter into the case pictures tool.

Minimises utilisation. Minimise letters.
### A.2. Stakeholders

<table>
<thead>
<tr>
<th>A-12</th>
<th>PAC</th>
<th>Maximise Fund Trustee value. Minimise operational costs.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PAC is an Administrator.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A digital form system is an electronic system that will process forms that are posted electronically. This includes via email, iPad app, or through a fund member portal. The pension company has recently moved into the realm of digital form systems where a member may use an online web form to make a request or to a change of details request this bypasses the whole mailing process and the need for a scanned picture so digital form system will create a digital form that will be stored in MyCasePictures and as PAC continues to evolve there will be less emphasis on MyCasePictures and the mailing unit as there as a demand to increase the number of digital forms used by members within the pension company.</td>
<td></td>
</tr>
</tbody>
</table>
The current system is based on the architecture as follows: Forms are all mailed in; they are scanned into an imaging system called MyCasePictures. The forms are then processed with OCR and work is tagged to a process and directed to queues. Staff must process forms based on business rules and process steps. Most processes are completed based on existing knowledge, and it is very common and possible for mail to be lost by staff members.

The new processes will accept both existing forms as well as digital forms, filled in on a smartphone, tablet or PC and emailed to the company. Instead of tagging forms to process and task queue, forms will now be tagged as belonging to a customer, and intelligent rules will be used to filter the form to a process and will allocate processing staff best suited to the customer to handle the task in an optimal way.

Process integration, compliance checking and strategy alignment are activities that will be conducted on the changed system to ensure conformance with legal and business goals as well as maximising the benefits of the system.

### A.3.1 Features and Benefits

Customer focused queues:

1. Personalised processing.
2. Easier customer prioritisation.
3. Best interaction guaranteed.

Digital Processing:

1. By opening up the digital space interaction with customers becomes more personal.
2. Faster responses for processors and customers.
3. More information available to processors and customers.

Process Integration:

1. Reduce complexity for processing staff.
2. All processes will follow the same steps, and business rules will automate decision making.
3. Free up processing staff to personalise service and offer alternative solutions.

A.3.2 External Requirements and Constraints

The system must be aligned with the corporate goals. There should be complete conformance with new legislation in the area.

A.3.3 Strategic alignment

This project supports strategy as follows:

1. Facilitates the delivery of objectives.
2. Is focused on the making the Customer the centre of operations.
3. Supports the employee career progression and development.
4. Developing future leaders through education and sustainable products.
5. Provides a richer digital platform for customer engagement.
A.3.4 Process Use Cases

A.3.4.0.1 UC-1 - Member Change Details When a member needs to update information, this process is triggered. Member information that may need to be updated includes any combination of:

1. Change of Name
2. Change of Address
3. Change of Contact Details
4. Change of Investment Options
5. Change of Beneficiaries (subprocess)

<table>
<thead>
<tr>
<th>Preconditions</th>
<th>Success Guarantee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current Membership Details</td>
<td>Member account is up to date.</td>
</tr>
<tr>
<td>Membership Number</td>
<td></td>
</tr>
<tr>
<td>First Name</td>
<td></td>
</tr>
<tr>
<td>Family Name</td>
<td></td>
</tr>
<tr>
<td>Date of Birth</td>
<td></td>
</tr>
<tr>
<td>Acknowledgment (Signature)</td>
<td></td>
</tr>
<tr>
<td>Signature dated</td>
<td></td>
</tr>
</tbody>
</table>

A.3.4.0.1.1 Main Success Scenario

1. Member initiates member change details request (either by phone or by mail)

2. The claim is redirected to the processors either from the contact centre or the mailing unit. In the new system, it is expected that the digital form system will also redirect requests to processors.

3. The member must supply identifying information to validate their claim. A processor or the digital form system will use validation business rules to determine valid supporting information.

4. Once verified the member must supply their updated information. The processor will then input these into the system and save the member account. For items
A.3. Product Overview

such as change of name (depending on validation business rules), certified proof is required. The member must send evidence of details in the form of certified proof before changes are finalised. If a member needs to post information to PAC then the Process Engine will pend the case subsequent to the pend business rules.

5. During the process, the Process Engine should check on the member business rules to check if any other information is required from the member. The Process Engine should also check if there is any extra information that can be supplied to the user about new products or services. This is done through using predictive member rules.

6. The member is transferred to any other processor that they need to speak to (as a result of further customer enquiries or business rules).

A.3.4.0.1.2 Extensions

<table>
<thead>
<tr>
<th>Related Requirements</th>
<th>Functional</th>
</tr>
</thead>
<tbody>
<tr>
<td>REQ-1</td>
<td>Fund specific (fund OLGST) change of details data fields</td>
</tr>
</tbody>
</table>

A.3.4.0.2 UC-2 - Member change beneficiary request  
A member may change their beneficiary and investment options using a number of methods. Beneficiaries are people associated with a member that will receive insurance or other benefits if the member is unable to claim their pension.

<table>
<thead>
<tr>
<th>Preconditions</th>
<th>Success Guarantee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Membership Number</td>
<td>Member beneficiary details have been updated</td>
</tr>
<tr>
<td>First Name</td>
<td></td>
</tr>
<tr>
<td>Family Name</td>
<td></td>
</tr>
<tr>
<td>Date of Birth</td>
<td></td>
</tr>
<tr>
<td>Acknowledgment (Signature)</td>
<td></td>
</tr>
<tr>
<td>Signature dated</td>
<td></td>
</tr>
</tbody>
</table>

A.3.4.0.2.1 Main Success Scenario
1. **Member** initiates **member** change beneficiary request (Either by Phone or by Mail)

2. The claim is redirected to the **processors** either from the contact centre or the mailing unit. In the new system, it is expected that the **digital form system** will also redirect requests to **processors**.

3. The **member** must supply identifying information to validate their claim. A **processor** or the **digital form system** will use validation business rules to determine valid supporting information.

4. Once verified the **member** must supply valid beneficiary details. The **processor** will then input these into the system and save the **member** account.

5. During the process, the **Process Engine** should check on the **member business rules** to check if any other information is required from the **member**. The **Process Engine** should also check if there is any extra information that can be supplied to the user about new products or services. This is done through using predictive member rules.

6. **The member** is transferred to any other **processor** that they need to speak to (as a result of further customer enquiries or business rules).

**A.3.4.0.2.2 Extensions**

**A.3.4.0.3 UC-3 - Member Consolidation**  
**Member** consolidation is the process of merging multiple accounts into one. Generally, consolidation is done between funds when a **member** switches pension fund.

<table>
<thead>
<tr>
<th>Preconditions</th>
<th>Success Guarantee</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Contact details are required (a mailing address and business name)</td>
<td>The member has all funds in one member account.</td>
</tr>
<tr>
<td>ETP</td>
<td></td>
</tr>
<tr>
<td>Personal Details</td>
<td></td>
</tr>
<tr>
<td>Certified Proof of Identification</td>
<td></td>
</tr>
</tbody>
</table>
A.3.4.0.3.1 Main Success Scenario

1. The member submits a form requesting that their account is consolidated.
2. The Processor will contact the member’s secondary fund account provider.
3. The secondary fund processor will acknowledge the request and confirm member details.
4. The secondary fund to submit a check or EFT with the member balance.
5. The processor will deposit the check and inform the Member

A.3.4.0.3.2 Extensions  1.a A new member may request that an account be opened to deposit pension funds.
2.a The pension fund processor may be contacted by a secondary fund requesting a rollover.
5.a If the member has not set their investment options, and then a request for investment choices is made.

A.3.4.0.4 UC-4 - Member Enrollment  This process is called when a customer wishes to join the fund as a new member. The member will request an account be created, and all initialisation details are set. This process can be initiated by a member or their employer.
Preconditions
Date of Birth
Name: First Name and Last Name of Customer
Home Address: Combination of Street Address + Town or Street Address + Postcode or Street Address + Postcode + Town)
Signature that has been dated within the last 6 months
Contact phone number
Signature + date
If investment options are selected than required (100%) allocation
If beneficiaries are selected, then require (100%) allocation

Success Guarantee
The fund has a new member, and the client has a member account.

A.3.4.0.4.1 Main Success Scenario
1. **Member** initiates joining process.
2. **Member** provides all details required, including certified proof of person and address.
3. Initial balance can be paid through ETP.
4. The processor takes all details and inputs into member business rules system. In the end, the member account is created.

A.3.4.0.4.2 Extensions

A.3.4.0.5 UC-5 - Employer Registration When a new employer wishes, to offer funding services as default to their employee’s they will register.
Preconditions
External Contact details are required (a mailing address and business name)

Success Guarantee
The employer will be enrolled on the PensionProcessSystem

A.3.4.0.5.1 Main Success Scenario

1. Employer initiates the process.

2. Processor reviews incoming form and verifies all details have been provided.

3. Processor reviews incoming documentation and certified proof to ensure that the employer is the right business.

4. The processor creates the employer on the PensionProcessSystem.

5. If the employer provides employee details, spreadsheet of employees is entered as new members.
A.3. Product Overview

A.3.5 Process Descriptions

Reconciliation Template

Model

1. Open Batch reconciliation template
2. Fill in cheques removed from batch section for all cheques that are removed from the batch
3. Fill in the cheques added to the batch section for all new cheques to batch
4. Compare the template total to the calculated total
5. After saving add the batch number generated by capital to the batch report
6. Separate cheques from paper work
7. Tally up all torn off cheques using printing calculator staple calculation total to cheque batch
8. Put all of the batch reports and cheques into a sleeve for banking

DB
Name Batch Reconciliation Template  
Type Human process  
Description Process for receipting user  
Scope  
Resources  
Governance Accounting Department  
Means  
Goals Ensure members contributions are allocated  
Actors Receptor (processor)  

<table>
<thead>
<tr>
<th>Business Rules</th>
<th>Rule Name</th>
<th>Rule Definition</th>
</tr>
</thead>
</table>
| PAC_BRT_R1     | MEMBER_CONTRIBUTION_RECEIPTED | <-
| CHEQUE_IN_BATCH AND TOTAL_VALUE_CORRECT |

<table>
<thead>
<tr>
<th>Activity</th>
<th>Name</th>
<th>Type</th>
<th>QOS Cost</th>
<th>Time</th>
<th>Skill Required</th>
<th>Utility</th>
<th>Effect</th>
<th>Constraint</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compare the template total to the calculated total</td>
<td>Human Task</td>
<td>$30</td>
<td>60 minutes</td>
<td>High</td>
<td>30</td>
<td>Total value is correct</td>
<td>Total value must be correct</td>
<td>Member contribution receipted</td>
<td></td>
</tr>
</tbody>
</table>
Activity
Name: Put all of the batch reports and cheques into a sleeve for banking
Type: Human Task
QOS Cost: $10
Time: 10 minutes
Skill Required: Low
Utility (times completed): 50

Effect: Cheque is in batch.

Receipting Contributions

Model

Receive batch of cheques and batch report
Cross reference cheques against Mailing Unit report one by one
Is there a problem with the batch report?
Return problem cheque to the Mailing Unit
Amend Batch report

Authenticate form
Verify Member
Verify member number in form
Receipt cheques to the members account
Complete Batch reconciliation template
Send the form to AGST member services to register the member
Separate forms, send cheque and batch report to the Mailing Unit
Send forms to other units for further processing
### Name
Receipting Contributions

### Type
Human process

### Description
Process for receipting user

### Scope

### Resources
Governance Accounting Department

### Means

### Goals
Ensure members contributions are allocated

### Actors
Receptor (processor)

### Business Rules

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Rule Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAC_BRT_R1</td>
<td>MEMBER_CONTRIBUTION_RECEIVED &lt;- CHEQUE_IN_BATCH AND TOTAL_VALUE_CORRECT</td>
</tr>
</tbody>
</table>

### Activity

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>QOS Cost</th>
<th>Time</th>
<th>Skill Required</th>
<th>Utility (times completed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compare the template total to the calculated total</td>
<td>Human Task</td>
<td>$30</td>
<td>60 minutes</td>
<td>High</td>
<td>30</td>
</tr>
</tbody>
</table>

**Effect**
Total value is correct

**Constraint**
Total value must be correct

**Goal**
Member contribution receipted
Activity
Name Put all of the batch reports and cheques into a sleeve for banking
Type Human Task
QOS Cost $10
Time 10 minutes
Skill Required Low
Utility (times completed) 50
Effect Cheque is in batch

Process Change in details (general overview)

Model

Name Change in details (general overview)
Type General Process
Description This process shows the general process flow of a change request as completed by a processor
Scope
Resources
Governance Processing team
Means
Goals Member details are kept up to date
Actors Processor
### Business Rules

**Rule Name** Rule Definition

<table>
<thead>
<tr>
<th>Rule Name</th>
<th>Rule Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAC_CIDA_R1</td>
<td>MAINTAIN_MEMBERDETAILS &lt;- MEMBER_UPDATED AND MEMBER AND FORM_REVIEWED_DATA_COMPLETE AND CRITICAL_DATA</td>
</tr>
</tbody>
</table>

### Activity

**Name** Check form for all critical data items

**Type** Human Task

**QOS Cost** $5

**Time** 10 minutes

**Skill Required** Low

**Utility (times completed)** 500

**Effect** Form review complete

**Critical data reviewed**

**Constraint** Critical data attached

**Goal** Maintain member details

### Activity

**Name** Update member process

**Type** Human Task

**QOS Cost** $5

**Time** 10 minutes

**Skill Required** Low

**Utility (times completed)** 450

**Effect** Submitted by MemberMember details updated

**Constraint** Submitted by a member

**Goal** Maintain member details
Activity
Name Error
Type End Event
QOS Cost $0
Time 0 minutes
Skill Required Low
Utility (times completed) 20

Effect Member details NOT updated and NOT maintained
Constraint
Goal

Activity
Name Member not found
Type Human Activity
QOS Cost $0
Time 0 minutes
Skill Required Low
Utility (times completed) 10

Effect Not a member
Constraint
Goal

Activity
Name Form critical error handler
Type Human Activity
QOS Cost $5
Time 10 minutes
Skill Required Low
Utility (times completed) 10

Effect Form submitted not correctMissing critical data
A.3. Product Overview

Consolidation allocation

Model

Name Consolidation allocation

Type Core Process

Description This process shows the general process flow of a member roll-in as completed by a processor

Scope

Resources

Governance Processing team

Means

Goals New digital core consolidation process created

Actors Processor

Business Rules

Rule Name Rule Definition

PAC_CONS_R1 MEMBER_HAS_FUNDS_ALLOCATED <- ALLOCATION_COMPLETE AND CONSOLIDATION_COMPLETE

PAC_CONS_R2 NEW_MEMBER_PROCESSED <- MEMBER AND ALLOCATION_LETTER_SENT

PAC_CONS_R3 DIGITAL_CORE_PROCESS <- NEW_MEMBER_PROCESSED AND MEMBER_HAS_FUNDS_ALLOCATED
### A.3. Product Overview

<table>
<thead>
<tr>
<th>Activity</th>
<th>Name Check that the payer existing in PensionProcessSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$20</td>
</tr>
<tr>
<td>Time</td>
<td>15 minutes</td>
</tr>
<tr>
<td>Skill Required</td>
<td>Low</td>
</tr>
<tr>
<td>Utility (times completed)</td>
<td>200</td>
</tr>
</tbody>
</table>

**Effect**

Customer is a member

**Constraint**

Goal Member has funds allocated to their account

New membership processed

<table>
<thead>
<tr>
<th>Activity</th>
<th>Name Confirm the details on ETP and batch report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$10</td>
</tr>
<tr>
<td>Time</td>
<td>20 minutes</td>
</tr>
<tr>
<td>Skill Required</td>
<td>High</td>
</tr>
<tr>
<td>Utility (times completed)</td>
<td>400</td>
</tr>
</tbody>
</table>

**Effect**

Customer is a member

**Constraint**

Goal Member has funds allocated to their account

New membership processed
<table>
<thead>
<tr>
<th>Activity</th>
<th>Name Form is opened by a separate AGT member services team member and authorised</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$20</td>
</tr>
<tr>
<td>Time</td>
<td>30 minutes</td>
</tr>
<tr>
<td>Skill</td>
<td>Required High</td>
</tr>
<tr>
<td>Utility</td>
<td>(times completed) 200</td>
</tr>
<tr>
<td>Effect</td>
<td>Allocation Letter Created</td>
</tr>
<tr>
<td>Consolidation</td>
<td>Filed</td>
</tr>
<tr>
<td>Constraint</td>
<td>Member</td>
</tr>
<tr>
<td>Goal</td>
<td>Member has funds allocated to their account</td>
</tr>
<tr>
<td>New membership processed</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activity</th>
<th>Name Print 2 allocation letters - 1 with FUND header and 1 with PAC header. FUND confirmation letter is sent to customer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$40</td>
</tr>
<tr>
<td>Time</td>
<td>60 minutes</td>
</tr>
<tr>
<td>Skill</td>
<td>Required Med</td>
</tr>
<tr>
<td>Utility</td>
<td>(times completed) 400</td>
</tr>
<tr>
<td>Effect</td>
<td>Allocation Letter Created</td>
</tr>
<tr>
<td>Consolidation</td>
<td>Filed</td>
</tr>
<tr>
<td>Constraint</td>
<td>Member</td>
</tr>
<tr>
<td>Goal</td>
<td>Member has funds allocated to their account</td>
</tr>
<tr>
<td>New membership processed</td>
<td></td>
</tr>
</tbody>
</table>
Activity
Name Send letter to customer
Type Human Task
QOS Cost $20
Time 15 minutes
Skill Required Med
Utility (times completed) 600

Effect Allocation of consolidation complete
Constraint Member
Goal Member has funds allocated to their account
New membership processed

Mailing Unit Processing Generic Model
### A.3. Product Overview

<table>
<thead>
<tr>
<th>Name</th>
<th>Mailing Unit Processing Generic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Core Process</td>
</tr>
<tr>
<td>Description</td>
<td>This process shows the general process flow of the mailing unit</td>
</tr>
</tbody>
</table>

#### Scope

- Resources
- Governance: Mailing Unit
- Means
- Goals
- Actors: Mailing Unit

#### Activity

<table>
<thead>
<tr>
<th>Name</th>
<th>Add form to MyCasePictures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$7</td>
</tr>
<tr>
<td>Time</td>
<td>1 minutes</td>
</tr>
<tr>
<td>Skill Required</td>
<td>Low</td>
</tr>
<tr>
<td>Utility (times completed)</td>
<td>2000</td>
</tr>
</tbody>
</table>

Effect: Forms are processed paperlessly

#### Constraint

- Goal

---

<table>
<thead>
<tr>
<th>Name</th>
<th>Add cheque to Batch report</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$3</td>
</tr>
<tr>
<td>Time</td>
<td>10 minutes</td>
</tr>
<tr>
<td>Skill Required</td>
<td>High</td>
</tr>
<tr>
<td>Utility (times completed)</td>
<td>900</td>
</tr>
</tbody>
</table>

Effect: Cheque is attached to case

---

#### Constraint

- Goal
A.3. Product Overview

New Employer Application Complete Process

Model

Name New Employer Application Complete Process
Type Core Process
Description This process shows the process flow for the creation of a new employer
Scope
Resources
Governance Processing
Means
Goals Standardised Employer Process
Actors Processor

Business Rules
Rule Name Rule Definition
PAC_CIDA_R1 MAINTAIN_EMPLOYERDETAILS <- EMPLOYER AND FORM_REVIEWED_DATA_COMPLETE AND CRITICAL_DATA
### Activity
**Name**: Check form for all critical data items  
**Type**: Human Task  
**QoS Cost**: $5  
**Time**: 10 minutes  
**Skill Required**: Low  
**Utility (times completed)**: 500

**Effect**: Form review complete  
**Critical data reviewed**  
**Constraint**: Critical data attached  
**Goal**: Maintain member details

### Activity
**Name**: Create employer process  
**Type**: Human Task  
**QoS Cost**: $5  
**Time**: 10 minutes  
**Skill Required**: Low  
**Utility (times completed)**: 450

**Effect**: Submitted by employer  
**New employer created**  
**Constraint**  
**Goal**: Maintain employer details
Activity
Name Employer found
Type Human Activity
QOS Cost $0
Time 0 minutes
Skill Required Low
Utility (times completed) 10

Effect Employer already registered
Constraint
Goal

Activity
Name Form critical error handler
Type Human Activity
QOS Cost $5
Time 10 minutes
Skill Required Low
Utility (times completed) 10

Effect Form submitted not correctMissing critical data
Constraint
Goal
New Member Application Complete Process

Model

Name New Member Application Complete Process
Type Core Process
Description This process shows the process flow for the creation of a new member
Scope
Resources
Governance Processing
Means
Goals Standardised Member Processes
Akers Processor

Business Rules
Rule Name Rule Definition
PAC_CIDA_R1 MAINTAIN_MEMBERDETAILS MEMBER AND FORM REVIEWED_DATA COMPLETE AND CRITICAL_DATA
<table>
<thead>
<tr>
<th>Activity</th>
<th>Name Check form for all critical data items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$5</td>
</tr>
<tr>
<td>Time</td>
<td>10 minutes</td>
</tr>
<tr>
<td>Skill Required</td>
<td>Low</td>
</tr>
<tr>
<td>Utility (times completed)</td>
<td>500</td>
</tr>
</tbody>
</table>

Effect Form review complete  
Critical data reviewed  
Constraint Critical data attached  
Goal Maintain member details

<table>
<thead>
<tr>
<th>Activity</th>
<th>Name Create member process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Human Task</td>
</tr>
<tr>
<td>QOS Cost</td>
<td>$15</td>
</tr>
<tr>
<td>Time</td>
<td>25 minutes</td>
</tr>
<tr>
<td>Skill Required</td>
<td>Low</td>
</tr>
<tr>
<td>Utility (times completed)</td>
<td>450</td>
</tr>
</tbody>
</table>

Effect Submitted by non-MemberMember created in system/  
Constraint  
Goal Maintain member details
### Activity
Name: Member found  
**Type:** Human Activity  
**QOS Cost:** $0  
**Time:** 0 minutes  
**Skill Required:** Low  
**Utility (times completed):** 10

**Effect:** Member (already)  
**Constraint**  
**Goal**

---

### Activity
Name: Form critical error handler  
**Type:** Human Activity  
**QOS Cost:** $5  
**Time:** 10 minutes  
**Skill Required:** Low  
**Utility (times completed):** 10

**Effect:** Form submitted not correct  
**Missing critical data**  
**Constraint**  
**Goal**
Pend Case

Model

### Name Pending Generic
- **Type**: Human process
- **Description**: Process for pending a case

### Scope
- **Resources**: 
- **Governance Processing**: 
- **Means**: 
- **Goals**: Manage workflow tasks

### Actors
- Processor
Activity
Name Count Number of times pended
Type Human Task
QOS Cost $5
Time 1 minutes
Skill Required Low
Utility (times completed) 1500

Effect Case is paperless
Workflow tasks are managed correctly
Constraint
Goal Manage workflow tasks
A.3.6 Process Requirements

<table>
<thead>
<tr>
<th>Functional</th>
<th>Priority 1</th>
<th>Proposed</th>
</tr>
</thead>
</table>

A.3.6.0.6 REQ-1 Fund specific (fund OLGST) change of details data fields  The change of details process should be able to handle forms that have the following fields:
- Current Membership Details, Membership Number, First Name, Family Name, Date of Birth, New Contact Details, Home or Work Phone, Mobile, Home or Postal Address, Suburb/Town, State/Territory, Postcode, Email Address, New Name, First Name, Last Name, Certified Evidence, Explanation, Nomination of Beneficiaries (must add up to 100%), First Name, Family Name, Relationship, Share %, Correct Date of Birth, Date of Birth, Certified Proof of Birth, Reason for changing, Correct Eligible Service Date, New Date, Evidence of Change, Reason for changing, Cancelling Direct Debit, Arrangement, Check box, Tax file number, Acknowledgment (Signature), Signature dated

**Rationale:** These are fields that are fields that are currently in use on all OLGST change of details forms.

**Customer Satisfaction:** Trustee has a contractual obligation to satisfy storage and to process all of the above fields when handling OLGST change of details forms.

**Customer Dissatisfaction:** We would like to offer Trustee and their clients an alternative subset of all fields for all funds. This will be to standardise customer databases and also to minimise differences between processes.

**Priority:** HIGH

**Conflicts:** We need to breach full constraints on the use of all fields to minimise differences between new processes.

**History:** Discussed with client.

----

A.3.6.0.7 REQ-2 Change of details data fields integration  **Description:** To complete the transition to a new Process Engine system, existing processes, forms and data must be integrated as best as possible.

There is a requirement from PACs Trustee clients that the Process Engine can process
forms with the specific client information. Due to the way that various Trustees work, some forms are rare and the client would like to ensure that a single form can be used to do multiple tasks. PAC must map each form to the correct process and develop business rules that will collect the correct information.

**Rationale**: This requirement is in place due to SLA and other contractual requirements that PAC has in place with its customers.

**Customer satisfaction**: If an adequate set of rules can be created to map forms to the correct processes and data elements to correct data dictionary spots then the client will be very happy.

**Customer dissatisfaction**: If Trustee contracts are violated Trustee’s will be unhappy.

<table>
<thead>
<tr>
<th>Functional</th>
<th>Priority 1</th>
<th>Proposed</th>
</tr>
</thead>
</table>

A.3.6.0.8 **REQ-3 Fund specific (fund PTAST) change of details data fields** The change of details process should be able to handle forms that have the following fields:

Current Membership Details, Pension Number, Date of Birth, Name of Employer Fund Name, Title
Family Name, Given Name, Old Postal Address, Old Suburb, Old Postcode, New Postal Address, New Suburb
New Postcode, Contact telephone number, Signature, Date

**Rationale**: These are fields that are currently in use on all PTAST change of details forms.

**Customer Satisfaction**: Trustee has a contractual obligation to satisfy storage and processing all of the above fields when handling PTAST change of details forms. PTAST is primarily a pension fund provider and mainly deals with change of address cases on their forms. Other details are not common.

**Customer Dissatisfaction**: We would like to offer Trustee and their clients an alternative subset of all fields for all funds. This will be to standardise customer databases and also to minimise differences between processes.

**Priority**: Medium

**Conflicts**: We need to breach full constraints on the use of all fields to minimise differences between new processes.

**History**: Discussed with client.
A.3.6.0.9 REQ-4 Fund specific (fund PenServices) change of details data fields
The change of details process should be able to handle forms that have the following fields:
Current Membership Details, Pension Number, Date of Birth, Name of Employer, Fund Name, Title
Family Name, Given Name, Old Postal Address, Old Suburb, Old Postcode, New Postal Address, New Suburb
New Postcode, Contact telephone number, Signature, Date
Rationale: These are fields that are currently in use on all PenServices change of details forms.
Customer Satisfaction: Trustee has a contractual obligation to satisfy storage and processing all of the above fields when handling PTAST change of details forms. PenServices is primarily a pension fund provider and mainly deals with change of address cases on their forms. Other details are not common.
Customer Dissatisfaction: We would like to offer Trustee and their clients an alternative subset of all fields for all funds. This will be to standardise customer databases and also to minimise differences between processes.
Priority: Medium
Conflicts: We need to breach full constraints on the use of all fields to minimise differences between new processes.
History: Discussed with client.

A.3.6.0.10 REQ-5 Fund specific (fund PEFS) change of details data fields
The change of details process should be able to handle forms that have the following fields:
Current Membership Details, Membership Number, Title, Gender, Date of Birth, Family Name, Given Name
Postal Address, Suburb, State, Postcode, Daytime Phone Number, Email Address, New Contact Details, Membership Number, Title, Gender, Date of Birth, Family Name, Given Name, Postal Address, Suburb, State, Postcode, Daytime Phone Number, Email Address, Acknowledgment (Signature), Signature dated
A.3. Product Overview

Rationale: These are fields that are fields that are currently in use on all PEFS change of details forms.

Customer Satisfaction: Trustee has a contractual obligation to satisfy storage and processing all of the above fields when handling PEFS change of details forms.

Customer Dissatisfaction: We would like to offer Trustee and their clients an alternative subset of all fields for all funds. This will be to standardise customer databases and also to minimise differences between processes.

Priority: HIGH

Conflicts: We need to breach full constraints on the use of all fields to minimise differences between new processes.

History: Discussed with client.

<table>
<thead>
<tr>
<th>Functional</th>
<th>Priority 1</th>
<th>Proposed</th>
</tr>
</thead>
</table>

A.3.6.0.11 REQ-6 Fund specific (fund TPN) change of details data fields  
The change of details process should be able to handle forms that have the following fields:
Current Membership Details, Title, Date of Birth, Membership Number, Given Name, Surname, Home Phone Number
Work Phone Number, Mobile Phone Number, Email Address, (Home/Postal), Previous Address, Previous Suburb
Previous Country, Previous State, Previous Postcode, Change of Address, Home / Postal, Address, Suburb, Country
State, Postcode, Change of Name, Previous Surname, Previous Given Names, Title, change of preferred Beneficiaries
Surname, Given Name, Relationship, Percentage % Total Percentages must = 100%, Declaration, Acknowledgment (Signature) Signature dated

Rationale: These are fields that are fields that are currently in use on all TPN change of details forms.

Customer Satisfaction: Trustee has a contractual obligation to satisfy storage and processing all of the above fields when handling TPN change of details forms. TPN is primarily a lost pension provider and mainly deals with members who need to roll their pension into another account. Change of details form are not common.

Customer Dissatisfaction: We would like to offer Trustee and their clients an alternative subset of all fields for all funds. This will be to standardise customer databases and also to minimise differences between processes.
A.3. Product Overview

**Priority**: Medium

**Conflicts**: We need to breach full constraints on the use of all fields to minimise differences between new processes.

**History**: Discussed with client.

<table>
<thead>
<tr>
<th>Functional</th>
<th>Priority 1</th>
<th>Proposed</th>
</tr>
</thead>
</table>

**A.3.6.0.12 REQ-7 Fund specific (fund SECOND) change of details data fields**  
The change of details process should be able to handle forms that have the following fields:

- Current Membership Details
- Membership Number
- New Member Details
- New Surname
- New Given Names
- Title
- Date of Birth
- New Address
- State
- Postcode
- New telephone (Home)
- New telephone (Work)
- New Mobile Number
- New Email address
- Old Contact Details
- Previous Surname
- Previous Given Names
- Title
- Date of Birth
- Previous Address
- State
- Postcode
- Change of preferred Beneficiaries
- Surname
- Given Name
- Relationship
- Percentage % * Total Percentages must = 100%
- Change of Investment Choices
- Investment Options and Percentage %
- Change in Insurance Cover
- Section 1 (Check Box)
- Section 2 (Number)
- Section Three (Check Box)
- Declaration
- Marketing Check Box
- Acknowledgment (Signature)
- Signature dated

**Rationale**: These are fields that are fields that are currently in use on all SECOND change of details forms.

**Customer Satisfaction**: Trustee has a contractual obligation to satisfy storage and processing all of the above fields when handling SECOND change of details forms.

**Customer Dissatisfaction**: We would like to offer Trustee and their clients an alternative subset of all fields for all funds. This will be to standardise customer databases and also to minimise differences between processes.

**Priority**: HIGH

**Conflicts**: We need to breach full constraints on the use of all fields to minimise differences between new processes.

**History**: Discussed with client.
Bibliography


Integration: Method and Analysis. In Proceedings of the sixth on Asia-Pacific
conference on conceptual modelling, APCCM, pages 29–38. Australian Computer

[148] Evan D. Morrison, Aditya K Ghose, Hoa Khanh Dam, Kerry G Hinge, and
Konstantin Hoesch-Klohe. Strategic Alignment of Business Processes. Service-

approaches to workflow modeling and verification. In Logics for Emerging Appli-

[150] N Mundbrod, J Kolb, and M Reichert. Towards a system support of collabora-
tive knowledge work. In Business Process Management Workshops, volume 132
of Lecture Notes in Business Information Processing, pages 31–42. Springer Berlin
Heidelberg, 2013.

requirements: a process-oriented approach. IEEE Transactions on Software Engi-

[152] John Mylopoulos, Lawrence Chung, and Eric Yu. From object-oriented to goal-

In Proceedings of the 29th International Conference on Software Engineering, ICSE,

[154] M. Netjes, S. L. Netjes, H. A. Reijers, and W. M. P. van der Aalst. An Evolution-
ary Approach for Business Process Redesign - Towards an Intelligent System.
In Proceedings of the Ninth International Conference on Enterprise Information Sys-

the Sixth International Conference on Entity-Relationship Approach, pages 439–449,


[214] Alain Wegmann, Anders Kotsalainen, Lionel Matthey, Gil Regev, and Alain Giannattasio. Augmenting the Zachman Enterprise Architecture Framework with


