2010

Putting the architecting back into software architecture with systems thinking agent-based modelling

Trevor Harrison
University of South Australia

Allan Peter Campbell
University of Wollongong, pcampbel@uow.edu.au

Thong Nguyen
Defence Science and Technology Organisation, Australia

Publication Details

Harrison, T., Campbell, A. P. & Nguyen, T. (2010). Putting the architecting back into software architecture with systems thinking agent-based modelling. Systems Engineering and Test and Evaluation Conference (pp. 1-12). Adelaide:

Putting the architecting back into software architecture with systems thinking agent-based modelling

Abstract
This paper details exploratory research which treats architecting as a system. This human architecting system has a structure composed of decisions, interdependencies amongst decisions, decision making, decision makers and the decision-making environment. Agent-based modelling is used to model the architecting system, and simulation is used to visualise system behaviour over time. The goal is to map legitimate / optimal speed of architectural decision-making to an architecting system behaviour pattern. Knowing the appropriate behaviour pattern of early architecture evolution will provide a mechanism for fine-grained progress tracking of architectural design. Divergence from this behaviour pattern should provide early warning signs of an incomplete or overdone architecture.

Keywords
putting, architecting, back, into, software, architecture, systems, thinking, agent, modelling

Disciplines
Engineering | Physical Sciences and Mathematics

Publication Details
Harrison, T., Campbell, A. P. & Nguyen, T. (2010). Putting the architecting back into software architecture with systems thinking agent-based modelling. Systems Engineering and Test and Evaluation Conference (pp. 1-12). Adelaide:

SETE 2010

Meeting the Challenges of Introducing Complex Capability

Putting the Architecting Back into Software Architecture with Systems Thinking and Agent-Based Modelling

Systems Engineering Test and Evaluation Conference 2010

3-6 May, Adelaide

Trevor Harrison[1], Prof. Peter Campbell[1], Dr. Thong Nguyen[2]

[1]Defence and Systems Institute, [2]Defence Science and Technology Organisation

Contact: trevor.harrison@postgrads.unisa.edu.au

Abstract. This paper details exploratory research which treats architecting as a system. This human architecting system has a structure composed of decisions, interdependencies amongst decisions, decision making, decision makers and the decision-making environment. Agent-based modelling is used to model the architecting system, and simulation is used to visualise system behaviour over time. The goal is to map legitimate / optimal speed of architectural decision-making to an architecting system behaviour pattern. Knowing the appropriate behaviour pattern of early architecture evolution will provide a mechanism for fine-grained progress tracking of architectural design. Divergence from this behaviour pattern should provide early warning signs of an incomplete or overdone architecture.

INTRODUCTION

Background. The importance of any software or system architecture is well known. For example, architecture is the predominating mechanism for achieving desired overarching system-wide qualities (Clements02, p.19). What is not well known though, is for a time period allocated to a project for architectural design, what is the legitimate (humanly possible) progress of architecting? For the architect, can architectural design be complete within the time period? For a design review or evaluation of the architecture, could the architecture be complete given the time period available for architecting?

A finer-grained composition of architecture comes from viewing architecture as emanating from a set of decisions, and from viewing architecting as decision making. This paper builds on earlier research into software architecture decisions (Babar07), (Avgeriou07) by adding the recognition of the time dimension of decision making found in three areas of systems engineering; (1) cognitive systems engineering and heuristics for factors in variation of speed of decision making / architecting, (2) complexity science utilised in very large systems and system of systems to model this "messy, trial and error" design activity, (3) systems engineering usage of social sciences and social theory. Social theory is needed to bring numerous concepts together for a more holistic problem setting and problem definition before commencing modelling and theory envisioning.

The remainder of this paper is structured as follows: a RELATED WORK section provides i) illumination of a "dark and tangled" area of research inherent to studies of decisions and decision making taken from (Langley95), ii) approaches to handling these phenomena by current researchers of architecture decisions, iii) alternatives approaches proposed by this paper's authors. The following METHODS section greatly expands on these alternative approaches. The effectiveness-to-date of workarounds is documented in the RESULTS section. Finally, the next steps are documented in CONCLUSION.

RELATED WORK

Architecture Decisions. There is a dark and

tangled area of research when it comes to architecting of software systems. Software architecture decisions have mostly been researched over the last decade without any reference to decision making (Harrison09). Those indispensable factors affecting individual and team decision making within a project environment (Shore08) seem to have been dispensed with. An analysis of the field study by (Curtis87) covering large 'live' / in-situ software development projects in the USA found i) design to be a "messy, trial and error thing" for even the best designers, and ii) no such research of large in-situ projects has been carried out since (Glass03). (To date, this paper's authors have been unable to find a similar field study of large in-situ projects conducting software systems design.) Furthermore, a review of decision science literature by (Langley95) states the dilemma faced by all researchers of decision making; compromising a research method, or, compromising research results.

Decision Science. The advice of (Sterman00, p.90) is always model a problem, never model a system. Accordingly, we turn to an extensive review of decision science by (Langley95) which reveals four troublesome phenomena afflicting any kind of research into decisions and decision making. These troublesome phenomena are [1] intangible nature of decisions and the subsequent difficulties of identification, [2] no theory of decision-to-decision relationships and their interactions exists, [3] two opposed philosophies of decision making exist – rational and naturalistic, [4] there can never be a common / universal decision-making process.

These troublesome phenomena were used as a characterisation of the problem before commencing modelling, per Sterman's advice.

[PHENOMENA 1] Intangible Nature of Decisions and Subsequent Difficulties of Identification.

"While the concept of 'decision' itself (which we take to mean commitment to action) may imply distinct, identifiable choice, in fact many decisions cannot easily be pinned down, in time or in place." (Langley95, p.261)

"Looking back on their projects, most architects interviewed for the University of Southern California program conclude that the key choices they made were rarely obvious choices at the time." (Maier09, p.271)

The majority of software architecture decision research to-date adopts an architecture knowledge management (AKM) strategy (Babar09). The focus is on capturing associated decision rationale for tangible decisions. Documenting the rationale of a tangible decision may trigger recollections of preceding intangible decisions. AKM attempts to make decisions more explicit by awareness techniques.

The disciplines of Product Management and Enterprise Architecture increase the number of tangible decisions by making explicit a large number of non-technical architecture-impacting and architecture-influencing decisions. For example, a "Thinking Breakdown Structure" template provides a set of decisions which serves as a high level decision roadmap for an entire product development project (Fitch99, p.41). (The roadmap provides an analysis plan that helps avoid the common pitfalls of "diving into detail" and "analysis paralysis".)

Identification of as many decisions as possible is important for gauging volume of decision making, before and during architecting. The tactic for identification of intangible decisions discussed in this paper relies on decision inter-dependencies exhibited by decision-to-decision relationships. However, such relationships turn out to be the next troublesome area of research into decisions decision making.

[PHENOMENA 2] No Theory of Decision-to-Decision Relationships and Interaction Exists.

"The difficulty of using 'decision' as a primary unit of analysis is further aggravated by another phenomenon, sometimes acknowledged but rarely explicitly investigated: that decisions interact with one another." (Langley95, p.270)

Minimalist relationships between architecture decisions have always existed. For example, the hierarchical quality attributes models of (McCall77) and ISO-9126 (Zhu05, p.45) highlight relationships based on the structure of the hierarchy of quality-related decisions. The relationships between quality attributes are more complicated than hierarchical models can

describe. Relational models such as (Perry87) and (Gillies97) describe direct, inverse and neutral types of relationships. For example, "Integrity versus Efficiency" has an inverse relationship; the control of data access will need additional code, leading to a longer runtime and more storage requirements (Zhu05, p.29).

Specifically for architectures, three level hierarchy of architecture decisions have been created by (Florentz07), (Bredemeyer05). Position in the hierarchy dictates a level of abstraction of an architecture decision (Bredemeyer05, p.3).

The most extensive list of decision-to-decision relationships is to be found in an ontology of software architecture decisions (Kruchten04). Twelve different types of decision-to-decision relationships are given (Kruchten04, Table2). Visualisation of this ontology has recently been attempted by (Lee08) using the Prefuse tool (Heer05). Decision structure is visualised as graphs, in which decisions are represented as nodes and the decision relationships are the edges (Lee08, p.48).

Evolution (state transition from Kruchten's ontology) of individual decisions over time has also been visualised in (Lee08, Fig.3).

The time dimension of decision evolution is of great importance when tracking the progress of architecting. Priority / early architecture design decisions (Reinsertion97) (Bass05) exist and therefore appropriate timing of decision making is needed (in addition to appropriate time period). Technology roadmaps and forecasts (DeGregorio00) show decisions have a timeline or life-span, meaning decisions have to re-evolve after a period of time. Iterative development provides the opportunity revisit decisions that were made with limited choices or only a single choice.

It is well known in systems thinking that humans are better at coping with detailed complexity, but very poor at coping with dynamic complexity (Senge94, pp.71-72). In the METHODS section, similar visualisation to (Lee08) is used to cope with dynamic complexity of decision-to-decision relationships. Pictures work better for this than words because all parts of a picture can be seen at once. Animation is added to such visualisation, via simulation. Decision-to-decision interactions (and decision evolution) can thus be watched over a period of time.

[PHENOMENA 3] Two Opposed Philosophies of Decision Making Exist (1) Rational (deliberate), and (2) Naturalistic (instinctive).

"To summarize, the literature of organization decision making has tended to accept a dichotomy between rather clear and focused sequential processes on one end and rather dark and tangled 'anarchic' processes on the other." (Langley95, p.263)

"Truly successful decision making relies on a balance between deliberate and instinctive thinking." (Gladwell05, p.141)

One article on software architecture decisions which gives equal coverage of both rational decision making (RDM) and naturalistic decision making (NDM) philosophies is (Zannier07). The (Zannier07) article, like the Gladwell quote above, claim both RDM and NDM styles are used by architects.

Despite signs of NDM utilised in day-to-day software architecting, for example the ubiquitous use of design patterns (Maier09, p.41) (Meier09), authors of this paper have had great difficulty finding any literature of software architecture decisions with coverage of the NDM style. An overview of the field of software architecture decision research by (Barbar07) covers seven different groups of researchers whose research efforts had been ongoing for an average 10 year period. All seven research groups focus on RDM. Yet there needs to be a balance of RDM and NDM styles.

Fortunately, NDM is well recognised in Systems Engineering as heuristics. Architectural design heuristics are documented in (Maier09). The legitimate effect of heuristics on speeding up or slowing down decision making is found in cognitive systems engineering (Rasmussen94, p.65).

The sensitivities of timing of and time period for decision making have good coverage in NDM – significant for those priority / early architecture decisions already mentioned in the previous [PHENOMENA 2]. Furthermore, various individual, social / project and organisational factors in the speed of decision making within any project environment are also well documented (Whittingham09, p.2). Accounting for these project environment variables – within which architecting takes place – and adding

them to the RDM-dominated view of software architecture decision making is effectively new exploratory research. The technique being used to explore the affects of so many project environment variables on the speed of architectural decision-making, without running hundreds of live projects, is agent-based modelling and simulation (see METHODS section). Agent-based modelling and simulation is also the very same technique that is used for portraying an equal balance of RDM and RDM styles.

[PHENOMENA 4] No Common Decision-Making Process.

"'decision' and decision process as decomposable elements tend to become mere figments of the researchers' conceptions, or artefacts of their methods. Or to use an even more graphic metaphor, if a decision is like a wave breaking over the shore - that is, perhaps identifiable at some sort of climax - then tracing a decision process back into an organization becomes much like tracing the origin of a wave back into the ocean." (Langley95, p.264)

"The safest conclusion to be drawn from descriptive studies is that there is no single decision-making (or problem-solving) process. Mental activities vary enormously across individuals, situations and tasks." (Hodgkinson08, p.467)

(These quotes should be no surprise given the previous [PHENOMENA 3] which identified two opposing philosophies of decision making.) One book on software engineering decisions which has tackled this decision making process variation is (Hoover09). It's approach is to focus on improving the quality of inputs to any decision-making process, leaving the decision-making process itself well alone. Missing from (Hoover09) is any coverage of architecture and design decisions though.

Agent based modelling is useful for ill-defined processes (Lavery08). The behaviour of the system to be modelled is automata-centred as opposed to process centric (Lavery08, p.1). The ill-defined process to be modelled here with agent-based modelling is the decision-making process.

To assist with modelling of agent behaviour we us utilise systems thinking (Meadows08) (Senge94), in particular system dynamics (Meadows08, Ch.1). The 'system' being studied here is a socio-technical architecting system which encompasses a decision making process. We use behaviour-over-time graphs i.e. system archetypes (Senge94, Ch.6) to learn whether the system is approaching a goal or limit, and if so, how quickly. The goal of this architecting system is to finalise (fully evolve) all those decisions which constitute a software system architecture. The limit would be the time period available for architecture and architecture-related decision making.

METHODS

Synopsis. Table 1 summarises the previous section in a traceability matrix between troublesome phenomena that afflict research into decisions and decision making, and problem-solving approaches taken by researchers. This section expands much more on the third column.

Phenomena	Existing Approach(es)	Our Approach
[1] Intangible Decisions	Rationale Capture & Management	Decision Discovery from Relationships
[2] Non-existent Theory for Decision Relationships	Decision Hierarchies Relational Models Taxonomy	Decision Hierarchy System Dynamics Simulation
[3] Opposed Decision-Making Philosophies	RDM, or, NDM (mostly RDM)	RDM and NDM
[4] Process Variation (no common decision-making process)	Improve Process Inputs	Agent-based Modelling to Improve Throughput of Process Output

Table 1: Traceability from Phenomena Afflicting Research of Decision-Making to Problem Solving Approach

Use of System Dynamics. System dynamics models seek to characterise a problem dynamically, unfolding over time, which show how the problem arose and how it might evolve in the future (Sterman00, p.90).

There is a requirement to visualise the dynamism of decision-to-decision interactions as a technique for coping with dynamic complexity (as described earlier under [PHENOMENA 2]). The dynamism is caused by many factors, for example, the interdependencies and interactions amongst decisions. This systemic structure (built by decision-to-decision relationships) affects behaviour of the set of decisions as a whole. Individual decisions evolve. Collectively, the evolution of all architecture decisions and architecture-related decisions represent the evolution of an architectural design. Simulation is used to animate the work of (Lee08) on decision visualisation – see next sub-section.

What speeds up and what slows down decision making can be viewed with 'stocks' and 'flows' from System Dynamics (Sterman00, Ch.6). Stocks are the decisions and flows are the decision evolution (state transition) via one of human decision making, a decision's own behaviour, changes to the decision-making environment, or decision-to-decision interactions.

Influence diagrams (a.k.a. cause and effect diagrams) (Sherwood02) have been used to see the whole picture of decision-to-decision interactions. This technique has been particularly useful in documenting the many architecture-related decision to architecture decision interactions. (A source of many software architecture-related decisions and their affects on architecture can be found in (Wilson01)). In these types of interactions, cause and effect are distant in space and time, adding to further dynamic complexity.

Agent-based Modelling. A novel idea here is to use 'decisions' as agents. (Traditionally, agents make decisions.) Agent behaviour and interactions with other agents is largely driven by relationships - a much more rich arrangement of decision-to-decision relationships than other research-to-date.

Agent-based Modelling (ABM) has been chosen as a practical way of re-creating as close to a live, in-situ project environment as possible. This is the environment where architectural decision-making takes place. Such an environment has a plethora of individual, team, project, organisation and business milieu factors (Shore08) which influence speed and re-work of decision making i.e. decision evolution. The agent-based modelling and simulation tool being used is Repast-Simphony v1.2 (Argonne08).

Continuing with the black box view of decision-making processes found in (Hoover09) (to mitigate the effects of an infinite number of possible decision-making processes described earlier in [PHENOMENA 4]) this paper concentrates on the quality of outputs from agent-based modelling and simulation of a human architecting system. This paper's authors intend to concentrate on the throughput of decision making i.e. appropriate timely outputs. (Cockburn08) and (Harrison08) have both proposed utilising using Cumulative Flow Diagrams (CFDs) (Anderson04) for tracking throughput of a decision-making process. State transitions from (Kruchten04) could be considered as processing stages for a CFD. (At the time of writing, this is still under investigation.)

The worth of simulation is to be able to "run decision making (decision evolution) several hundred times". This is equivalent to running the same project several hundred times, with different project environment variables set on each run. By comparing all the sets of outputs, it should be possible to see which architecting system had the best throughput of decision making and what were the optimal project environmental variables that were part of the systemic structure of that architecting system.

Validation. Researchers of RDM have been more concerned with internal validity, whereas NDM researchers have been more concerned with external validity (Schnedier03, p.574). By definition, NDM research is focussed on decision makers in their natural settings, so it is not surprising that that NDM researchers stress the importance of ecological validity of field studies over the need for methodological rigour of surveys and experiments where conditions are under strict control.

Validation of simultaneous RDM and NDM within a synthetic, simulated project environment (a single simulation) poses a dilemma. This paper therefore adopts the position of (Sterman00); no model has ever been or ever will be thoroughly validated. "Useful", "illuminating", "convincing" or "inspiring confidence" are more apt descriptors applying to models than "valid" (Sterman00, p.846).

RESULTS

Ensemble of Decisions (large population). In order to derive general theories about life, we need an ensemble of instances to generalize over (North07, p.14). Likewise, to generalise a theory about architectural decision-making progress, we need an ensemble of decisions to generalise over. Hence the inclusion of both architecture and architecture-related decisions.

Systems thinking is famed for seeing the whole (Senge94, Ch.8). Sticking with this mindset, several hundred common decision agents (not just a few "significant decisions") are arranged into a grid representing a three level hierarchy of meta-, high-, and low-level decisions based on (Florentz07) and (Bredemeyer05) hierarchical views of software architecture decisions. Both architecture and architecture-related decisions are represented. The anatomy of an individual decision agent is shown in Table 2. Note that one alternative for one decision represents one agent. If there are three alternatives for one decision, then three individual decision agents exist in this model. (If two of the 3 alternatives are non-suitable "non-starters" very early on, their evolution is going to be short lived.)

Location in the hierarchy level alone indicates some of a decision agent's behaviour. For example, meta-level decisions influence (inform) high-level decisions, which may impact (tightly constrain) low-level decisions.

Property / Attribute	Description
decisionID	One of Mnnn, Hnnn or Lnnn.
epitome	Choose / Decide / Select <the decision itself>.
mustCrtieria	Mandatory criteria
wantCriteria	Desirable criteria
evolutionState	Idle | Need Idea | Idea | Proposed | Approved | Challenged | Obsolete
qualityPair	Paired decision
timeLine	Life span of decision
choice	One possible option

Table 2: Common Anatomy of Meta-, High- and Low-level Decision Agents

Decision Discovery. Because decisions are intangible (see [PHENOMENA 1] in "RELATED WORK" section), the ensemble of decisions (the "decision grid") is the starting point for discovery of related decisions. Figure 1 illustrates the two stages of decision discovery. The first stage is an equivalence match from a known tangible decision to a decision in the decision grid. The second stage of decision discovery relies on various types of relationships to/from the matched decision in the decision grid. Types of relationships include – but not limited to – trade-off pairs, parent-child influences, parent-child impacts and neighbouring decisions. Table 3 shows show neighbour decisions of one meta-level decision agent. Table 4 shows neighbour decisions of a low-level decision agent. ('Neighbours' meaning this cluster of decisions should be considered at the same time, or, are considered at different times by the same person/team.)

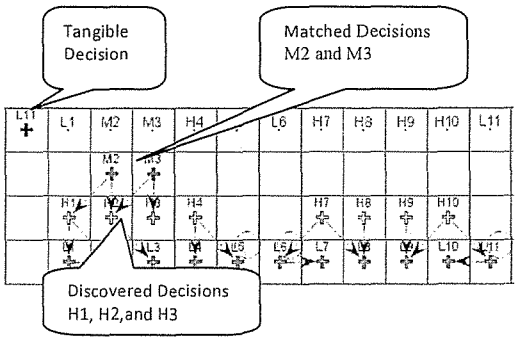

Figure 1. Discovering Decisions in a Decision Grid

Meta-level Decision	Neighbour Decisions
Decide system purpose.	Choose metaphor / organising concept.
	Choose system boundary(ies).
	State policy on development re-usable architecture assets.
	State policy on implementation re-usable architecture assets.
	Choose pervasive design paradigm(s).
	Choose pervasive implementation paradigm(s).
	Choose consumers of the architecture.
	Choose application archetype.
	Decide deployment infrastructure constraints.
	Decide development environment constraints.
	Choose value(s) the system will provide.

Table 3: Neighbouring Decisions of a Meta-level Decision

Low-level Decision	Neighbour Decisions
Choose data and message contracts that represent the schema for messages.	Define the service contracts that represent operations supported by the service.
	Decide how to handle invalid requests.
	Choose how to detect & manage duplicate messages.
	Decide how to handle messages out of order.
	Choose authentication strategy.
	Choose authorisation strategy.
	Choose communications protocol between a service and its consumers.
	Choose abstraction approach to interface with the Business Logic layer.

Table 4: Neighbouring Decisions of a Low-level Decision

Figure 2 shows those discovered decisions distilled out of the decision grid in Figure 1 ("projected") into a Repast-Simphony network layout.

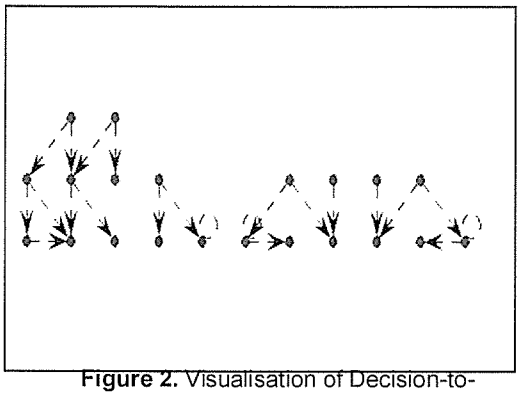

Figure 2. Visualisation of Decision-to-Decision Relationships with Repast-Simphony

Using the simulation engine of Repast-Simphony, the network graph in Figure 2 is to be animated to show the evolution of decisions. Similar to the visualisation by (Lee08), size of a node is dependent on the evolution state of that decision, as per the (Kruchten04) ontology.

Decision Evolution Over Time. Decision evolution of several decisions, visualised over a period of time by (Lee08) has been reproduced with permission in Figure 3.

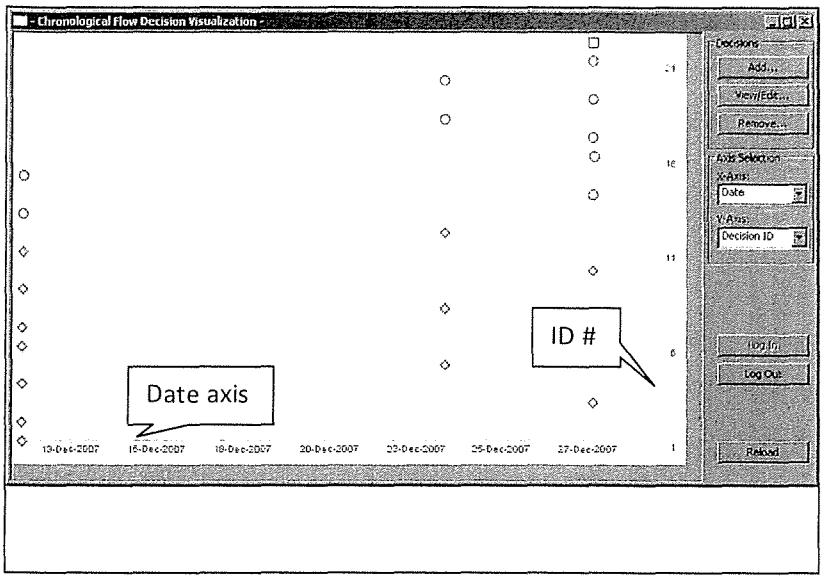

Figure 3. Decision Chronology (Lee08)

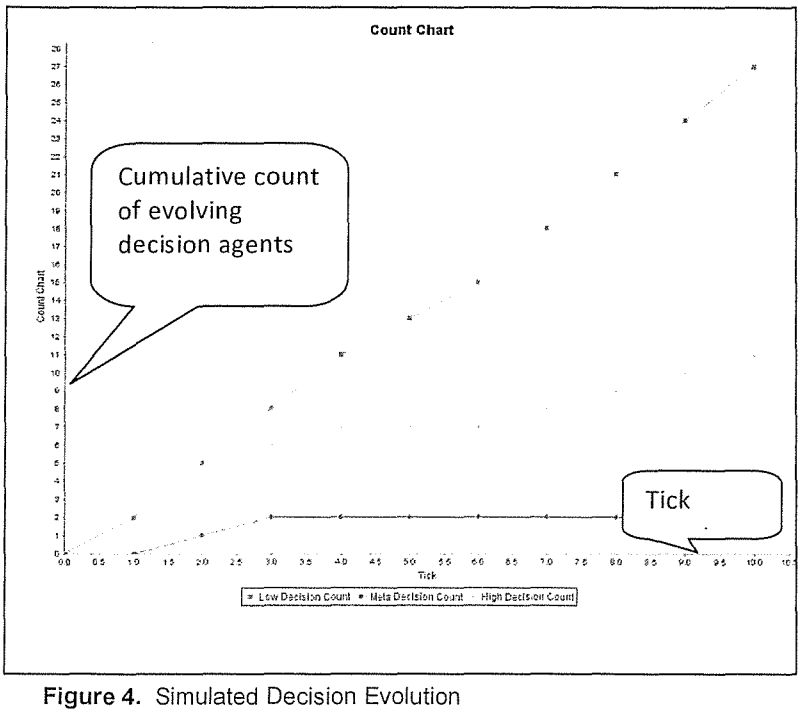

Figure 4. Simulated Decision Evolution

The view in Figure 3 is a fixed view; a snapshot of decision states at-a-moment-in-time. Figure 4 illustrates the 'graph' functions of Repast-Simphony, now showing evolution (state transition) of several decisions over finite intervals of time (the "tick count" on the horizontal axis). The evolution shown is focussed on decision discovery; an evolution from the state 'Idle' to 'Idea'. This can easily be replayed over and over again through Repast-Simphony's video capture functionality

Figure 4 illustrates behaviour of the human architecting system as a whole (per systems thinking), rather than the behaviour of individual parts. The sample behaviour would appear to be initially bottom-up design because low-level decisions are evolved / discovered first. Early evolutions of low-level decisions is consistent with observations of architecture design progress in a cyclic or episodic fashion; an architect's design role is not restricted to "high-level" considerations. Architects dig down into specific subsystem and domain details where necessary to establish feasibility (Maier09, p.254).

Findings from Simulations of Decision Discovery and Subsequent Evolution. At the time of writing, different decision matches and decision discoveries are being carried out. This is to determine which fragments / properties of which tangible decision discover the greatest number of related intangible decisions.

Discovery of a large set of decisions (amongst the decision grid) that constitute an architecture is just the very beginning step of a simulation in decision evolution / state transition. At the time of writing, random time periods for each state transition, for each decision are being assigned. This is so that a simulation engine can be established and simulation reports can be explored further.

CONCLUSIONS

The use of systems thinking has allowed decisions, decision making, decision makers and the decision-making environment to be viewed and analysed as a human architecting system.

Agent-based modelling and simulation has been used, and continues to be used to study the evolution of both individual decisions, and a set of decisions constituting an architecture. Simulation has permitted animation to be added to existing work on visualisation of aggregation of architecture decision evolution. Both are seen as helpful in coping with dynamic complexity.

The opening INTRODUCTION section stated the importance of the right amount time required for architecting. This paper's approach has acknowledged this time period will vary depending on the decision-making capability / throughput of individuals and teams, and the complexity of the architecture design itself. A distribution of variations still needs to be produced by agent-based modelling and simulation. The upper and lower bounds of these distributions then need to be cross-checked in discussions with architects, and where available, cross-checked once more with project documentation or project reviews.

An inappropriate time period for all decision making, or inappropriate timing of decision making of certain decisions, will always result in the same outcome; re-work and re-testing.

REFERENCES

(Anderson04) Anderson. David, "*Using Cumulative Flow Diagrams*", Agile Management, May 2004 edition, http://dn.codegear.com/article/32410 , accessed September 2007.

(Argonne08) Argonne National Laboratory, *Recursive Porous Agent Simulation Toolkit (REPAST)*, http://repast.sourceforge.net/ , 2008, accessed Jan 2010.

(Avgeriou07) Avgeriou, Paris et al, *Architectural Knowledge and Rationale – Issues, Trends, Challenges*, ACM SIGSOFT Software Engineering Notes, Volume 32 Number 4, pp41-46, July 2007.

(Babar07) Babar, Muhammad Ali et al, *Architectural Knowledge Management Strategies: Approaches in Research and Industry*, Second workshop on SHAring and Reusing architectural Knowledge (SHARK '07), 2007.

(Barbar09) Babar, Muhammad Ali et al, *Software Architecture Knowledge Management: Theory and Practice*, Springer, 2009.

(Bass05) Bass, Len, *Principles of Software Architecture Evaluation and Design*, Microsoft Asia-Pacific Architect Forum, Gold Coast, Australia, 2005.

(Bredemeyer05) Bredemeyer, Ruth and Dalen, Bredemeyer, *Software Architecture: Central Concerns,Key Decisions*, http://www.bredemeyer.com/pdf_files/ArchitectureDefinition.PDF Bredemeyer Consulting, 2005.

(Clements02) Clements, Paul, Rick Kazman and Mark Klein, *Evaluating Software Architectures : Methods and Case Studies*, Addison-Wesley, USA, 2002.

(Cockburn08) Cockburn, Alistair, *Lean Processes: Humans and Technology*, gantthead.com article, http://www.gantthead.com, July 7, 2008, accessed January 2010.

(Curtis87) Curtis, Bill, Herb Krasner and Neil Iscoe, "*A Field Study of the Software Design Process for Large Systems*", Communications of the ACM, November 1988, Volume 31, Number 11.

(DeGregorio00) DeGregorio, Gary, *Technology management via a set of dynamically linked roadmaps*, pp. 184-190, Proc. IEEE EMS Int Eng Management Conf., Albuquerque, USA, August 2000,

(Fitch99) Fitch, John, *Structured Decision-Making & Risk Management*, Student Course Notes, Systems Process Inc., 1999.

(Florentz07) Florentz, B., and Huhn, M., *Architecture Potential Analysis: A Closer Look inside Architecture Evaluation*, Journal of Software, Vol. 2, No. 4, October 2007.

(Gladwell05) Gladwell, Malcolm (2005), *Blink: The Power of Thinking without Thinking*, Penguin Books, 2005.

(Glass03) Glass, Robert L., "*Facts and Fallacies of Software Engineering*", Addison-Wesley, Boston, 2003.

(Gillies97) *Software Quality: Theory and Management*, 2nd edition, International Thompson Computer Press, 1997.

(Harrison08) Harrison, Trevor and Nguyen, Thong, *Determining System Quality Attributes from the Architectural Decision-Making Process*, Qualcon 2008, Adelaide, Australia.

(Harrison09) Harrison, Trevor et al, *Design Uncertainty Theory - Evaluating Software System Architecture Completeness by Evaluating the Speed of Decision Making*, Proceedings from the 1st Improving Software and Systems Engineering Capability conference 2009, Canberra, Australia.

(Heer05) Heer, J., Card, S.K., Landay, J.A., *Prefuse: a toolkit for interactive information visualization*, SIGCHI conference on Human factors in computing systems , 2005.

(Hodgkinson08) Hodgkinson, Gerald P., and Starbuck, William H., *The Oxford Handbook of Organizational Decision*

Making, Oxford University Press, USA, 2008.

(Hoover09) Hoover, Carol L., *Evaluating Project Decisions: Case Studies in Software Engineering*, Addison-Wesley, 2009.

(Kruchten04) Kruchten, Philippe, *An Ontology of Architectural Design Decisions in Software Intensive Systems*, Proc. of the 2nd Workshop on Software Variability Management, Groningen, NL, Dec. 3-4, 2004.

(Langley95) Langley, Ann et al, *Opening up Decision Making: The View from the Black Stool*, Organization Science, Vol. 6, No. 3, May-June 1995.

(Lavery08) Lavery, Eamon, *Introduction to Agent Based Simulation in Flexsim*, http://www.flexsim.com/products/ds/FlexsimABS.pdf accessed November 2009.

(Lee08) Lee, Larix and Kruchten, Philippe (2008), *A Tool to Visualize Architectural Design Decisions*, QoSA 2008, Lecture Notes in Computer Science, pp. 43–54, Springer-Verlag.

(Maier09) Maier, Mark W., and Rechtin, Eberhardt, *The Art of Systems Architecting*, Third Edition, CRC Press, 2009.

(Meadows08) Meadows, Donella H., *Thinking in Systems : A Primer*, Chelsea Green Publishing Company, USA, 2008.

(Meier09) Meier, J.D., *Microsoft Application Architecture Guide : patterns and practices*, 2nd edition, Microsoft Press, 2009

(North07) Michael J. North and Charles M. Macal, *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*, Oxford University Press, 2007.

(Patterson93) Patterson, Marvin and Sam Lightman, "*Accelerating Innovation*", Van Nostrand Reinhold, New York, USA, 1993.

(Perry87) Perry, W., *Effective Methods of EDP Quality Assurance*, 2nd edition, Prentice-Hall, 1987.

(Rasmussen94) Rasmussen, Jens, Annelise Mark Pejtersen, and L.P. Goodstein (1994), *Cognitive Systems Engineering*, Wiley-Interscience, 1994.

(Reinersten97)Reinersten, Donald G., *Managing The Design Factory – A Product Developers Toolkit,* Free Press, 1997.

(Senge94) Senge, Peter, *The Fifth Discipline*, 2nd revised edition, Random House Books, 1994.

(Schnedier03) *Emerging Perspectives on Judgement and Decision Research*, Cambridge University Press.

(Sherwood02) Sherwood, Dennis, *Seeing the Forrest for the Trees: A Manager's Guide to Applying Systems Thinking*, Nicholas Brealey Publishing, London, 2002.

(Shore08) Shore, Barry, *Systematic Biases and Culture in Project Failures*, Project Management Journal, December 2008, Vol.39, No. 4, pp.5-16.

(Sterman00) Sterman, John D., *Business Dynamics: Systems Thinking and Modeling for a Complex World*, McGraw Hill, USA, 2000.

(Whittingham09) Whittingham, Ian, *Hubris and Happenstance: Why Projects Fail*, 30th March 2009, `http://gantthead.com` , 2009.

(Wilson01) Wilson, James R., *Software Architecture: Organizational Principles and Patterns*, Prentice-Hall, 2001.

(Zannier05) Zannier, Carmen, and Maurer, Frank, *A Qualitative Empirical Evaluation of Design Decisions*, Human and Social Factors of Software Engineering (HSSE) May 16, 2005, St. Louis, Missouri, USA.

(Zannier07) Zannier, Carmen, Mike Chiasson and Frank Maurer, *A model of design decision making based on empirical results of interviews with software designers*, Information and Software Technology Journal no. 49, pp.637-653, 2007.

(Zhu05) Zhu, Hong, *Software Design Methodology: from principles to architectural styles*, Elsevier, 2005.

BIOGRAPHY

Trevor Harrison's research interests are in software systems architecture. His background is in software development (real-time information systems), technology change management and software engineering process improvement. Before studying full-time for a PhD, he spent 6 years with Logica and 11 years with the Motorola Australia Software Centre. He has a BSc(Hons) in Information Systems from Staffordshire University and an MBA (TechMgt) from La Trobe University.

Prof. Peter Campbell is the Professor of Systems Modelling and Simulation and Research Leader in the Defence and Systems Institute (DASI) at the University of South Australia. In 2004 Prof. Campbell was the founding member of the Centre of Excellence for Defence and Industry Systems Capability (CEDISC). Both organisations have a focus on up-skilling government and defence industry in complex systems engineering and systems integration. He currently leads the design for the simulation component of the DSTO MOD funded Microcosm program and is program director for three other DSTO funded complex system simulation projects. Through mid 2007, he consulted to CSIRO Complex Systems Science Centre to introduce complex system simulation tools to support economic planning of agricultural landscapes.

Dr Thong Nguyen Dr Nguyen holds BE (Honour I), BSc and PhD from Adelaide University, and MBA (TechMgt) from Deakin University. He is currently the Research Program Manager for Airborne Mission Systems Branch, AOD, where he has been leading a team of AMS researchers, NICTA and UniSA colleagues to develop AMS's expertise on Capability Integration which is a combination of Software Intensive Systems Integration and Information Integration. The research program employs different techniques such as AADL, Colour Petri Net, Queuing Theory, Bayesian Networks, Intelligent Agents, and software architecture evaluation methods to assess software qualities and integration at the architecture level. It also includes three aspects of Information Integration: Management, Fusion, and Exchange.