

Understanding Enterprise Behavior using Hybrid Simulation of Enterprise Architecture

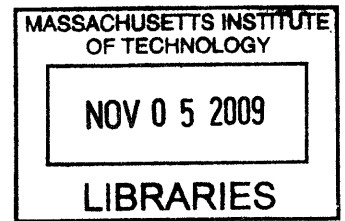
By
Christopher Garrett Glazner

B.S. Electrical Engineering – University of Texas at Austin, 2003
B.A. Plan II Honors – University of Texas at Austin, 2003
M.S. Technology and Policy – Massachusetts Institute of Technology, 2006

SUBMITTED TO THE ENGINEERING SYSTEMS DIVISION IN PARTIAL
FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY IN TECHNOLOGY, MANAGEMENT AND POLICY
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

ARCHIVES

[June 2009]
April 27, 2009



©2009 Massachusetts Institute of Technology. All Rights Reserved

Author.....
Engineering Systems Division

Certified by.....
Deborah Nightingale, PhD
Professor of Aerospace and Astronautics and Engineering Systems
Thesis Supervisor

Certified by.....
Joseph M. Sussman, PhD
Professor of Civil and Environmental Engineering and Engineering Systems
Thesis Committee Member

Certified by.....
Kirkor Bozdogan, PhD
Principal Research Associate, Center for Technology, Policy and Industrial Development
Thesis Committee Member

Accepted by...
Nancy Levinson, PhD
Professor of Aerospace and Astronautics and Engineering Systems
Chair, Engineering Systems Division Education Committee

Understanding Enterprise Behavior using Hybrid Simulation of Enterprise Architecture

By
Christopher Garrett Glazner

Submitted to the Engineering Systems Division
in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in
Engineering Systems

ABSTRACT

Today, the design of business enterprises is much more art than science. The complex structure and behavior of enterprises makes it difficult to untangle cause and effect amidst its components and their relationships. In order for managers to understand how an enterprise's architecture affects its behavior, they need tools and techniques to help them to manage the complexity of the enterprise. The practice of enterprise architecting continues to make advances in this area with reference frameworks that can be used to guide the decomposition and communication of enterprise architectures, but it does not provide tools to analyze the potential behavior of a proposed enterprise architecture.

This research seeks to extend the practice of enterprise architecting by developing an approach for creating simulation models of enterprise architectures that can be used for analyzing the architectural factors affecting enterprise behavior and performance. This approach matches the content of each of the "views" of an enterprise architecture framework with a suitable simulation methodology such as discrete event modeling, agent based modeling, or system dynamics, and then integrates these individual simulations into a single hybrid simulation model. The resulting model is a powerful analysis tool that can be used for "what-if" behavioral analysis of enterprise architectures. This approach was applied to create a hybrid simulation model of the enterprise architecture of a real-world, large-scale aerospace enterprise. Simulation model analysis revealed potential misalignments between the current enterprise architecture and the established strategy of the enterprise. The simulation model was used to analyze enterprise behavior and suggest relatively minor changes to the enterprise architecture that could produce up to a 20% improvement in enterprise profitability without increasing resources to the enterprise.

Thesis Supervisor: Prof. Deborah Nightingale
Title: Professor of the Practice of Aerospace & Astronautics and Engineering Systems

For Summer

Acknowledgements

As with most worthwhile endeavors, this research would not have been possible without the help and support of a large ensemble of mentors, colleagues, and friends that have helped me develop and mature the thinking in this dissertation and motivate me through the process. First and foremost, I must thank my committee who have supported and encouraged me as I have worked to complete this dissertation for the past three years. My thanks to Prof. Nightingale, for her passion and creative thinking; to Prof. Sussman for his keen insight and sage advice, and to Dr. Bozdogan for his insatiable appetite for knowledge and for introducing me to the world of enterprise architecture. I am also greatly indebted to Geoff Bentley and Jim Anapol, who not only made the case study in this dissertation possible, but also contributed a number of ideas that enriched the development and presentation of the model. Without their help, support and guidance, this effort would have not been possible.

I also owe a debt of gratitude to my friends and colleagues at the Lean Advancement Initiative, who over the years have helped me develop my thinking in these areas. Among the many who have passed through 41-205, I would like to thank JK Srinivasan, Dr. Sgouris Sgouridis, Nirav Shah, John Dickmann, Dr. Heidi Davidz, Dr. Donna Rhodes, Dr. Ricardo Valerdi, Dr. Eric Rebintisch, Caroline Lamb, Maj.(Dr.) Jason Bartolemei, Dr. Marc Haddad, Phech Colatat, Tom Shields, Juliet Perdichezzi, and Tara Eisner. I would also like to thank several of my colleagues at the MITRE Corporation who have continued to help me refine my thinking on enterprise architecture and enterprise dynamics: Dr. Ken Hoffman, Dr. Len Wojick, Fran Dougherty, Dr. Richard Ryberg and Bill Bunting,

There are two professors in particular that I must thank for starting me down the path of engineering systems: Dr. Hans Mark and Dr. Billy Koen of the University of Texas at Austin. They planted in me the seeds of system thinking and the

desire to use my engineering education to bring about large-scale change. I will never again be the same.

Finally, I would like to thank my wife, Summer, for all of the countless things that she has done to help me through my graduate studies and through this dissertation. Without her, this entire process would have not begun nor would it have finished. 'Thank you' is not enough!

Table of Contents

<u>CHAPTER 1: INTRODUCTION</u>	19
1.1 THE ENTERPRISE AS A COMPLEX SYSTEM	21
1.2 ABSTRACTING ENTERPRISE COMPLEXITY	23
1.3 HYBRID SIMULATION MODELING	25
1.4 A PROPOSED ENTERPRISE ARCHITECTURE BASED HYBRID SIMULATION METHODOLOGY	26
1.5 RESEARCH OBJECTIVES	28
1.6 THESIS OUTLINE	29
<u>CHAPTER 2: MANAGING ENTERPRISE COMPLEXITY WITH ENTERPRISE ARCHITECTURE FRAMEWORKS AND ORGANIZATIONAL SCIENCE</u>	31
2.1 COMPLEXITY IN ENTERPRISES	32
2.1.1 A “SYSTEMS APPROACH” AND A “REDUCTIONIST APPROACH” OF COMPLEX SYSTEMS	34
2.1.2 NEAR-DECOMPOSABILITY OF COMPLEX SYSTEMS	35
2.1.3 SCALE IN COMPLEX SYSTEMS	37
2.1.4 PERSPECTIVE IN COMPLEX SYSTEMS: “TOP-DOWN” VERSUS “BOTTOM-UP”	38
2.1.5 APPLYING COMPLEXITY THEORY TO ENTERPRISES	40
2.2 ORGANIZATIONAL SCIENCE	42
2.2.1 THE OPEN SYSTEM PERSPECTIVE OF ORGANIZATIONS	43
2.2.2 CONTINGENCY THEORY	44
2.2.3 ORGANIZATIONAL DESIGN	45
2.2.4 COMPUTATIONAL ORGANIZATIONAL SCIENCE	52
2.2.5 GAPS IN ORGANIZATIONAL THEORY FOR EFFECTIVE ORGANIZATIONAL DESIGN	55
2.3 ENTERPRISE ARCHITECTING	56
2.3.1 DEVELOPMENT OF ENTERPRISE ARCHITECTURE FRAMEWORKS	58
2.3.2 ENTERPRISE ARCHITECTURE FRAMEWORK VIEWS	62
2.3.3 THE USE OF VIEWS IN POPULAR ENTERPRISE ARCHITECTURE FRAMEWORKS	66
2.3.4 INTERACTIONS AMONG VIEWS	71
2.3.5 THE NIGHTINGALE-RHODES ENTERPRISE ARCHITECTURE FRAMEWORK	73
2.3.6 APPLICATION OF ENTERPRISE ARCHITECTURE FRAMEWORKS FOR ENTERPRISE ARCHITECTING	76

2.3.7	UTILITY OF THE ENTERPRISE ARCHITECTURE FOR UNDERSTANDING AND MANAGING ENTERPRISE BEHAVIOR	78
-------	---	----

2.4 A SYNTHESIS OF ORGANIZATIONAL SCIENCE AND ENTERPRISE ARCHITECTURE FRAMEWORKS
83

2.4.1	THE NEED FOR SIMULATION MODELS	85
-------	--------------------------------	----

CHAPTER 3: APPROACHES TO SIMULATING ENTERPRISE ARCHITECTURE **87**

3.1 CURRENT MODELING METHODS USED TO SUPPORT ENTERPRISE ARCHITECTING **89**

3.1.1	EXAMPLES OF ENTERPRISE ARCHITECTURE MODELING APPROACHES IN PRACTICE	90
-------	---	----

3.1.2	THE SHORTCOMINGS OF CURRENT APPROACHES TO ENTERPRISE ARCHITECTURE MODELING	94
-------	--	----

3.2 WHY USE SIMULATION MODELS? **95**

3.3 WHAT DO ENTERPRISE LEADERS NEED FROM ENTERPRISE SIMULATION MODELS? **98**

3.3.1	REPRESENTATIVE OF THE ENTERPRISE	99
-------	----------------------------------	----

3.3.2	CAPTURES THE MECHANISMS UNDERLYING BEHAVIORAL COMPLEXITY	100
-------	--	-----

3.3.3	SPECIFIC AND TIMELY	101
-------	---------------------	-----

3.3.4	CAPABLE OF ADAPTATION	102
-------	-----------------------	-----

3.4 SIMULATION METHODOLOGIES FOR ENTERPRISE BEHAVIOR **102**

3.4.1	DISCRETE EVENT SIMULATION FOR THE ENTERPRISE	103
-------	--	-----

3.4.2	SYSTEM DYNAMICS MODELING	109
-------	--------------------------	-----

3.4.3	AGENT-BASED MODELING	115
-------	----------------------	-----

3.4.4	COMPARISON OF SIMULATION METHODOLOGIES FOR ENTERPRISE ARCHITECTING	125
-------	--	-----

3.5 HYBRID SIMULATION MODELING **128**

3.5.1	BACKGROUND OF HYBRID ENTERPRISE SIMULATIONS	129
-------	---	-----

3.6 CONCLUSIONS **131**

CHAPTER 4: SIMULATING ENTERPRISE ARCHITECTURE USING A HYBRID APPROACH **133**

4.1 PRINCIPLES FOR CREATING HYBRID SIMULATION MODELS OF THE ENTERPRISE ARCHITECTURE
134

4.1.1	MODELING FOR INSIGHT, NOT PREDICTION	135
-------	--------------------------------------	-----

4.1.2	MODELING THE ARCHITECTURE, NOT THE ENTERPRISE	136
-------	---	-----

4.1.3	STRATEGIC LEVEL MODELING OF THE ENTERPRISE ARCHITECTURE	137
-------	---	-----

4.1.4	FOCUS ON DYNAMICS RESULTING FROM BY INTERACTIONS ACROSS THE ARCHITECTURE	138
-------	--	-----

4.2 A PROPOSED HYBRID, ENTERPRISE ARCHITECTURE FRAMEWORK-BASED MODELING APPROACH	138
4.3 A PROCESS FOR DEVELOPING HYBRID MODELS OF THE ENTERPRISE	141
4.3.1 STEP 1: DOCUMENT THE ENTERPRISE ARCHITECTURE	144
4.3.2 STEP 2: PROBLEM ARTICULATION	145
4.3.3 FORM A DYNAMIC ARCHITECTURAL HYPOTHESIS	147
4.3.4 STEP 4: IDENTIFY THE APPLICABLE ARCHITECTURAL VIEWS	147
4.3.5 STEP 5: MATCH VIEWS WITH SIMULATION METHODOLOGIES	148
4.3.6 STEP 6: IDENTIFY BOUNDARIES AND INTERFACES FOR EACH VIEW	151
4.3.7 STEP 7: CREATE A SUB-SYSTEM AND HYBRID-LEVEL DIAGRAMS	155
4.3.8 STEP 8: IMPLEMENT THE HYBRID SIMULATION MODEL	159
4.3.9 MODEL TESTING	162
4.3.10 STEP 10: POLICY DESIGN AND EVALUATION	165
4.4 SUMMARY	166
<u>CHAPTER 5: THE TECHSYS CASE STUDY</u>	<u>169</u>
5.1 THE TECHSYS ENTERPRISE	170
5.2 CASE SELECTION	172
5.3 HISTORY OF TECHSYS	173
5.3.1 A NEW STRATEGY REQUIRES A NEW ENTERPRISE ARCHITECTURE	174
5.3.2 DEVELOPING COMMON PROCESSES AND METRICS	179
5.3.3 TECHSYS MAKES ITS ACQUISITIONS	180
5.3.4 TECHSYS'S CHALLENGES	182
<u>CHAPTER 6: THE APPLICATION OF THE FRAMEWORK FOR CREATING HYBRID ENTERPRISE ARCHITECTURE SIMULATIONS</u>	<u>185</u>
6.1 STEP 1: DOCUMENTING THE ENTERPRISE ARCHITECTURE	186
6.1.1 INITIAL DATA COLLECTION FOR EA DOCUMENTATION	187
6.1.2 ENTERPRISE ARCHITECTURE ITERATION	189
6.2 STEP 2: PROBLEM ARTICULATION	189
6.2.1 IDENTIFYING AND BOUNDING THE ROOT PROBLEM	191
6.2.2 IDENTIFYING INPUTS AND OUTPUTS	194
6.2.3 TIME HORIZON	197

6.2.4	KEY STRUCTURES AND BEHAVIORS	197
6.2.5	APPLICABILITY OF HYBRID ENTERPRISE ARCHITECTURE MODELING TO THE PROBLEM	198
6.3	STEP 3: FORM A DYNAMIC ARCHITECTURAL HYPOTHESIS	199
6.4	STEP 4: IDENTIFY THE APPLICABLE VIEWS FROM THE ENTERPRISE ARCHITECTURE	201
6.5	STEP 5: MATCH THE VIEWS WITH SIMULATION METHODOLOGIES	204
6.5.1	STRATEGY/FINANCE SUB-MODEL	205
6.5.2	ORGANIZATIONAL SUB-MODEL	206
6.5.3	PROCESS SUB-MODEL	206
6.5.4	MODELING OTHER VIEWS WITH VARIABLES	208
6.6	STEP 6: IDENTIFY BOUNDARIES AND INTERFACES BETWEEN THE VIEWS/SUB-MODELS	209
6.6.1	CREATING A LOGICAL INTERACTION DIAGRAM	210
6.6.2	BOUNDARY MODEL CHARTS	214
6.7	STEP 7: CREATE SUB-MODEL AND TOP-LEVEL MODEL DIAGRAMS	216
6.7.1	THE SUB-MODEL DIAGRAMS	217
6.7.2	HYBRID MODEL DIAGRAM	223
6.8	STEP 8: IMPLEMENT THE HYBRID SIMULATION MODEL	229
6.8.1	ACQUIRING DATA	230
6.8.2	SIMULATION SOFTWARE	233
6.8.3	IMPLEMENTING THE PROCESS SUB-MODEL	237
6.8.4	DEVELOPING THE USER INTERFACE	242
6.8.5	RUNNING THE TECHSYS SIMULATION MODEL	243
 <u>CHAPTER 7: TESTING AND ANALYSIS OF THE TECHSYS ENTERPRISE ARCHITECTURE</u>		
<u>SIMULATION MODEL</u>		<u>249</u>
7.1	TESTING THE TECHSYS SIMULATION MODEL	250
7.1.1	TESTING FOR BOUNDARY ADEQUACY AND A STRUCTURAL ASSESSMENT	251
7.1.2	PARAMETER AND VARIABLE ASSESSMENT	252
7.1.3	EXTREME CONDITION TESTING	255
7.1.4	SENSITIVITY ANALYSIS	258
7.1.5	SUMMARY OF MODEL TESTING	258
7.2	ANALYSIS USING THE TECHSYS SIMULATION MODEL	259
7.2.1	INVESTMENT IN PURSUING SYNERGY	260
7.2.2	ALLOCATING THE DISCRETIONARY BUDGET	265

7.2.3	COMBINING THE LEVERS: THE PERFORMANCE LANDSCAPE FOR THE CURRENT-STATE ENTERPRISE ARCHITECTURE	270
7.2.4	CREATING AN ALTERNATIVE ARCHITECTURE	272
7.2.5	PERFORMANCE OF THE ALTERNATIVE ENTERPRISE ARCHITECTURE	274
7.3	RECOMMENDATIONS FROM THE USE OF THE TECHSYS SIMULATION MODEL	279
7.3.1	OTHER LESSONS LEARNED FROM THE TECHSYS SIMULATION MODEL	282
7.3.2	THE BENEFITS OF A HYBRID APPROACH TO ENTERPRISE ARCHITECTURE SIMULATION MODELING	285
7.3.3	FUTURE APPLICATION OF THE SIMULATION MODEL AT TECHSYS	286
<u>CHAPTER 8: CONCLUSIONS, CONTRIBUTIONS, AND FUTURE WORK</u>		289
8.1	REVIEW OF THESIS	290
8.2	CONCLUSIONS	293
8.3	CONTRIBUTIONS	295
8.3.1	OTHER APPLICATIONS OF HYBRID ENTERPRISE ARCHITECTURE SIMULATION MODELS	298
8.4	DIRECTIONS FOR FUTURE WORK	298
8.5	CONCLUDING REMARKS	300
<u>WORKS CITED</u>		303

List of Tables

<i>Table 2-1: Table comparing characteristics associated with top-down and bottom-up approaches to systems analysis</i>	40
<i>Table 2-2: A table of views employed by popular enterprise architecture frameworks.</i>	68
<i>Table 2-3: The views employed by the Nightingale-Rhodes Enterprise Architecture Framework</i>	74
<i>Table 2-5: A comparison of strengths and weakness of current approaches to enterprise architecting as a framework for analysis of enterprise dynamics</i>	82
<i>Table 3-1: Commonly used Enterprise Architecture Toolsets</i>	92
<i>Table 3-2: A comparison of potential simulation methodologies</i>	124
<i>Table 4-1: Steps of the hybrid, EA-based simulation modeling process. Arrows indicate major feedback loops. Adapted from Sterman (2000).</i>	143
<i>Table 4-2: Classes of variables that can be used as interface variables for three different simulation methodologies</i>	154
<i>Table 4-3: An example Model Boundary Chart</i>	155
<i>Table 5-1: List of modifications to the TechSys Enterprise Architecture to align with new strategy</i>	182
<i>Table 6-1: Selection of simulation approaches for the sub-models</i>	207
<i>Table 6-2: Model Boundary Chart of the Strategy/Finance View sub-model</i>	215
<i>Table 6-3: Model Boundary Chart for the Organizational View sub-model</i>	215
<i>Table 6-4: Model Boundary Chart for the Process View sub-model</i>	216
<i>Table 6-5: Qualitative operating unit comparison variables and their valuation criteria</i>	233
<i>Table 6-6: Opportunity entity attributes</i>	238
<i>Table 7-1: Summary of the Alternative Architecture</i>	273

List of Figures

<i>Figure 2-1: An example of the structure of a nearly-decomposable system</i>	36
<i>Figure 2-2: Galbraith's Star Model (1973)</i>	46
<i>Figure 2-3: Teece's Dynamic Capabilities Framework (2007)</i>	51
<i>Figure 2-4: The cover image from Douglas Hofstadter's "Gödel, Escher, Bach"</i>	63
<i>Figure 2-5: Using an architecture framework with multiple views to reduce apparent complexity</i>	64
<i>Figure 2-6: A simplified Zachman Framework Matrix</i>	69
<i>Figure 2-7: The GERAM/ ISO 15704 cube.</i>	70
<i>Figure 2-8: The Views and Interactions in the Nightingale-Rhodes Enterprise Architecture Framework</i>	75
<i>Figure 3-1: Examples of the extremes of discrete event simulations. h.</i>	105
<i>Figure 3-2: A System Dynamic model of an inventory system with a single feedback loop</i>	110
<i>Figure 3-3: The Game of Life, three iterations of a basic "X"</i>	117
<i>Figure 3-4: The BOIDS model, showing flocking behavior of birds around obstacles.</i>	118
<i>Figure 3-5: Application of Simulation Methodologies on Abstraction Level scales.</i>	125
<i>Figure 4-1: A notional representation of a hybrid, enterprise-architecture based simulation model</i>	140
<i>Figure 4-2: A hybrid model centered on a discrete event process sub-model</i>	156
<i>Figure 4-3: A hybrid model centered on an agent based model</i>	157
<i>Figure 5-1: The organizational structure of BigTechs and TechSys</i>	171
<i>Figure 5-2: The organizational structure of TechSys after 2005</i>	177
<i>Figure 6-1: Conceptual "black box" level depiction of the TechSys Simulation Model</i>	196
<i>Figure 6-2: A conceptual diagram of relationships between architectural factors related to the pursuit and capture of new business opportunities</i>	197
<i>Figure 6-3: Logical Interaction diagram of the TechSys Enterprise Architecture Simulation Model</i>	211
<i>Figure 6-4: Sub-model Diagram of the Strategy Sub-model</i>	218
<i>Figure 6-5: Agent structure for the organizational sub-model</i>	219
<i>Figure 6-6: The sub-model diagram of the process view</i>	222
<i>Figure 6-7: A high-level structural diagram of an operating unit</i>	225
<i>Figure 6-8: Organizational sub-model diagram of an operating unit</i>	227
<i>Figure 6-9: Top-level diagram, showing input and output parameters</i>	228
<i>Figure 6-10: The operating unit sub-model in the AnyLogic development environment</i>	235
<i>Figure 6-11: Historical cost distribution for Level C opportunities at one TechSys operating unit.</i>	239
<i>Figure 6-12: The TechSys Enterprise Architecture Simulation Model Interface (single replication)</i>	241
<i>Figure 6-13: The histogram representing the combined profitability from organic and synergistic profits for a single simulation run of the current-state architecture.</i>	244

Figure 6-14: A graph of the moving mean of the cumulative profit from the TechSys model, using 2007 input parameters and the current architecture. _____ 246

Figure 6-15: A graph of the moving standard deviation of the cumulative profit from the TechSys model, using 2007 input parameters and the current architecture _____ 246

Figure 7-1: Distribution of the number of business opportunities evaluated in one year of the simulation for OU #1 for a single simulation run with 200 replications. _____ 254

Figure 7-2: Distribution of opportunity sizes produced by the model and from historical data. $R^2=.87$, Mean Absolute Error = 2% _____ 255

Figure 7-3: Normalized Profitability of TechSys as a function of allocation of the Discretionary budget to Bid and Proposal, with the remainder allocated to IRAD. _____ 257

Figure 7-4: Expected profit as synergy investment is varied _____ 260

Figure 7-5: The mechanics of Synergy and Organic Opportunity Selection _____ 262

Figure 7-6: Normalized Expected Profit as the allocation of the Discretionary Budget is varied between IRAD (0%) and Bid and Proposal (100%) _____ 266

Figure 7-7: Expected profits as the Discretionary budget allocation is varied, for time horizons of 3, 5, and 10 years. _____ 268

Figure 7-8: A simplified causal loop diagram of the dynamics of allocation of the Discretionary budget _____ 269

Figure 7-9: The performance landscape for the current state TechSys enterprise architecture _____ 271

Figure 7-10: Expected profitability versus percentage of new business opportunities that are synergistic, for both the current state and alternative architectures _____ 274

Figure 7-11: Expected Profits versus the percentage of the Discretionary budget allocated to Bid and Proposal, for both the current state and alternative architectures _____ 277

Figure 7-12: The performance landscape for the alternative enterprise architecture _____ 278

Figure 8-1: The proposed method for creating hybrid, enterprise architecture based simulation models _____ 296

Chapter 1: Introduction

Today, despite the efforts of researchers in many fields over the past half-century, the design and management of enterprises remains as much art as science. The complex structure and behavior of enterprises makes it difficult to untangle the relationship between form and behavior; changes to one aspect of an enterprise's structure, incentives, or strategy can affect its behavior in seemingly unrelated areas at distant points in time. An enterprise is composed of many different elements, such as the business plan, organizational structures, processes, and technologies, and all of these components interact, often in ways that are difficult for an individual or group with limited visibility and cognitive capacity to anticipate. Nonlinearities, inertia, delays, and feedback in the system all contribute to the difficulty in understanding how complex enterprise behaviors are influenced by the "design" of the enterprise that produced them. It is exceedingly difficult for enterprise leaders to anticipate how any changes to their enterprise's form may impact its behavior without tools to help them analyze the behavior and its drivers.

Imagine the task of a chief executive officer, faced with a shifting business environment challenging the existing business model. There are a host of questions that may be asked in such a situation:

- How can the enterprise be retooled to capitalize on a newly developed business model?
- Will a new business model require changes to the enterprise's organization, processes, or knowledge requirements?

- How can a new organizational form and incentives be developed that will be responsive to new customer demands or more quickly take advantage of new technologies in its products and processes?
- How could alternative proposed forms of the enterprise be compared to each other? How could tradeoffs be assessed?

Currently, enterprise leaders have few tools available that can help them wrestle with questions such as these that concern the enterprise's *architecture*. Most such major decisions concerning an enterprise's architecture are made without the aid of a tool that allows experimentation to test hypotheses. Previous practical efforts to develop tools and processes to conceive and analyze aspects of the enterprise, such as business process reengineering (Hammer and Champy, 1992) and value stream mapping (Womack and Jones, 1996) have relied heavily on "brown-paper walling" or "whiteboarding," employing Post-It™ notes and hand-drawn lines to convey new organizational structures and processes. More recent efforts have simply used digital versions of this static "boxes and lines" approach to modeling the enterprise's structure. Such approaches do not create "live models" that can be subjected to hands-on experimentation, however. What is required is the capability to bridge the gap between descriptive approaches to enterprise design and quantifiable models from which results can be collected and analyzed (Fowler, 2003). Without the ability to analyze new architectures, there cannot be a formal, reliable process for designing (or redesigning, or evolving) the enterprise (Levitt, 2004).

In an ideal situation, the CEO would have a trusted simulation model of the enterprise that could be used to test hypothesized design changes and compare alternatives and scenarios. Simulation models are intended to mimic the behavior of a real system, allowing that behavior to be studied in a controlled way. Using simulation models, a modeler can create a "microworld" that can be used to conduct experiments that are faster, more flexible, and allow the development of

policy options in a practical and more ethical manner¹ than real world experimentation would allow (Carley, 2002). The idealized, “crystal ball” simulation model would allow enterprise leaders to make any number of changes to an enterprise’s architecture in the model and then run the model over a period of time to assess how the changes may impact potential behavior and performance of the enterprise. Such a crystal ball would give an enterprise a considerable advantage by identifying advantageous forms for its business model and its desired behaviors.

1.1 THE ENTERPRISE AS A COMPLEX SYSTEM

Unfortunately, such an ideal, comprehensive model that captures every facet of an enterprise’s design and behavior would be extremely difficult, if not impossible to create in practice. Enterprises are very complex² systems—many orders of magnitude more complex than the largest models that modern computers can handle (Simon, 1990). Enterprises are both structurally complex, in that they have a great number of interconnections, as well as behaviorally complex, in that the behavior of the system cannot be understood or anticipated by study of the constituent parts—its behavior is more than the sum of its parts. There are many factors that make enterprises complex. Enterprises are:

- Socio-technical systems, combining “hard” technical elements as well as “soft,” cultural and organizational elements
- highly-interconnected with many feedback loops, inertia and delays;
- filled with autonomous people all making local and perhaps not-strictly-rational decisions;
- capable of adaptation; and
- embedded in a constantly shifting, open environment.

¹ Simulation experiments can be considered more ethical than performing experiments on functioning enterprises which may effect the livelihood of participants.

² Enterprise complexity will be defined and discussed in detail in Chapter 2.

The complexity of enterprises makes creating the ideal, crystal ball simulation model almost impossible. A predictive simulation of complex system that seeks to model all inputs, outputs, and interactions is a futile endeavor. The data for such models is often approximated, and the full nature of all interactions is not known. Enterprises exhibit chaotic behavior: they are stochastic and sensitive to small perturbations to their conditions that make accurate prediction close to impossible (Dooley and Van de Ven, 1999). Sterman argues "it is simply not possible to build a single, integrated model of [a complex system], into which mathematical inputs can be inserted and out of which will flow a coherent and useful understanding of world trends" (1991). A fully detailed, predictive model of a complex system would necessarily be as complex as the original system.

The key to modeling complex systems such as an enterprise is to properly abstract them. Modelers must abstract from the complexity of the enterprise in a way that helps to highlight critical interactions and relationships that drive behaviors of interest, while ignoring other interactions that do not contribute to a systems-level understanding of the enterprise. Herbert Simon argues that "intelligent approximation, not brute force computation, is still the key to effective modeling" (Simon, 1990). For many problems, the answers that are needed do not require a highly detailed, predictive model, but rather one that is capable of understanding general trends, paths, and steady states. Rather than focus on models that try to predict the chaotic behavior of enterprises, modelers and leaders should instead seek out "organization-specific generative models that can explain how the chaotic behavior came about in the first place" (Dooley and Van de Ven, 1999). Enterprises must be understood in terms of their dynamic behavior, and models must be designed to capture patterned sequences of events driven by the organizational design (Abbott 1990). Enterprise modelers should not concern themselves with how to build more detailed models, but instead how to develop abstractions of enterprises in a way that allows more insight into their behavior, and ultimately do a better job of conceiving and managing them.

1.2 ABSTRACTING ENTERPRISE COMPLEXITY

One approach to developing an abstraction of the enterprise from a complex systems perspective is the practice of *enterprise architecting*. Enterprise architecting holds that the enterprise can be both understood and designed by employing the construct of *enterprise architecture* as a unifying conceptual framework. The enterprise architecture is a documented abstraction of the fundamental organization of an enterprise as a dynamic holistic system with nonlinearly interacting components.³ The architecture of an enterprise is an abstraction of its essential features, rather than a complete, detailed description of its design. In order to help enterprise architects develop these abstractions, *enterprise architecture frameworks* can serve as a starting point in establishing scope and identifying key abstractions, boundaries, and interactions within an architecture. Each framework identifies multiple *views* that can be used to decompose the architecture from different perspectives, such as strategy, organizational structure, processes, and information technology. These views are interconnected together as a system. See Figure 1-1.

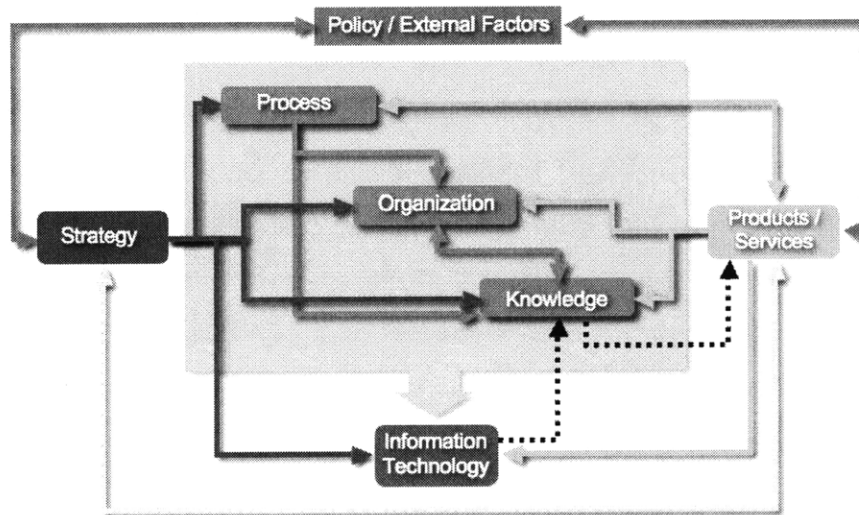


Figure 1-1: The views and interactions in the Nightingale-Rhodes Enterprise Architecture Framework

³ The concept of enterprise architecture will be addressed in detail in Chapter 2.

Unlike the theoretical literature on organizational design and architecture that remains highly fragmented and multifaceted, an important immediate benefit of the enterprise architecture reference frameworks is that they provide a practical way for defining and decomposing enterprises as an interconnected system, relating the disparate components of the architecture together. Enterprise architecture frameworks are intended to serve as a unifying platform for many different disciplines of enterprise design and study to allow their combined and coordinated application, rather than their individual, disjoint application. They do not provide any theoretically grounded guidance of their own to guide the architecting process, but instead unify the theories developed in disparate research areas such as organizational science, management science, and business process design in order to achieve this task. They provide well-defined workable representations of aspects of the enterprise that can then be analyzed by employing a number of methods to help understand and create enterprise architectures.

Enterprise architecture frameworks provide a potentially useful way of simplifying and abstracting an enterprise's complexity by decomposing it into its major constituent components, each representing a discrete view, and by linking them together into an interconnected whole, by utilizing associated theory. The resulting enterprise architectures, however, are static representations and present limited opportunity for quantitative analysis of the underlying dynamic enterprise architecture. It is thus difficult to use an enterprise architecture framework alone to answer a question concerning the enterprise's potential behavior. To answer such questions, a simulation model of enterprise behavior is needed. The enterprise architecture, however, may be used to provide the necessary abstractions, scoping, structure and interactions necessary to create a simulation model to answer the question.

1.3 HYBRID SIMULATION MODELING

The complexity of enterprises requires that they be abstracted into different subsystems in order to be understood. Each subsystem presents a new perspective on the enterprise, complete with its own context. For the modeler attempting to model behaviors this presents a challenge: many problematic behaviors span multiple perspectives, and there is no one single simulation modeling methodology that is capable of simulating each perspective in its own context (Mingers and Gill, 1997). The tools used to simulate the behavior of an incentivized collection of people within an organization are very different from those used to simulate the execution of a manufacturing process or top-down strategic resource allocation decisions. Each simulation methodology has its own strengths and weaknesses, based on the assumptions and mechanisms that it uses to simulate the system at hand. For this reason, it has been argued that a cross-disciplinary approach to simulation modeling of enterprises be taken, employing a portfolio of models from different fields and for different purposes, allowing the individual models to be compared, contrasted, and critiqued (Sterman, 1991).

Others have gone on to argue that using a portfolio of stand-alone simulation models does not accurately convey the system's dynamics, and that a hybrid, multi-methodology approach to simulation should be used (Mingers and Gill, 1997; Rabelo et. al., 2005). In a hybrid simulation model, multiple sub-models employing different simulation methodologies are interfaced with each other such that the execution of one sub-model can be used as the input of another, forming a system of interacting sub-systems. In recent years, this hybrid approach to modeling complex systems has gained traction in some areas of enterprise modeling, such as supply chains (Scheritz and Größler 2003; Rabelo et. al. 2007) production planning (Venkateswaran and Son 2005), and manufacturing decision-making (Rabelo, et. al. 2005). These hybrid simulation models have employed system dynamics, agent-based models, and discrete event simulations to capture different perspectives of a system, have applied each methodology to

areas where it is the best method to describe and understand the system's behavior.

To date, the few hybrid models of enterprise operations have been fairly modest in scope. There has not been a formalized approach to determining the boundaries between sub-models in the hybrid models or the pathways of their interactions. Previous hybrid modelers have taken an *ad hoc* approach to boundary setting between simulation sub-models. *Ad hoc* approaches will become increasingly problematic as the scope and complexity of hybrid models increases. Further, each time a new model is required for to capture a new aspect of the same enterprise's behavior, a new effort to abstract the portions of the enterprise being modeled is required. This can prove to be a very substantial amount of work, and may also provide obstacles to communicating the boundaries and scope of the model.

1.4 A PROPOSED ENTERPRISE ARCHITECTURE BASED HYBRID SIMULATION METHODOLOGY

Simulation models of enterprise behavior can take advantage of an initial definition of the existing enterprise architecture in order for them to capture an adequate abstraction of the enterprise as it exists and thus guide their boundaries. Using such an approach, the views of the enterprise architecture can be used to define the boundaries of the sub-models. Interactions among sub-models can then be modeled using the inter-view interactions in the enterprise architecture, by employing known theoretical propositions. The use of the existing enterprise architecture, as a reference framework, ensures that the hybrid model is consistent with the developed architecture of the enterprise and permits the model to be more easily understood and communicated to enterprise stakeholders.

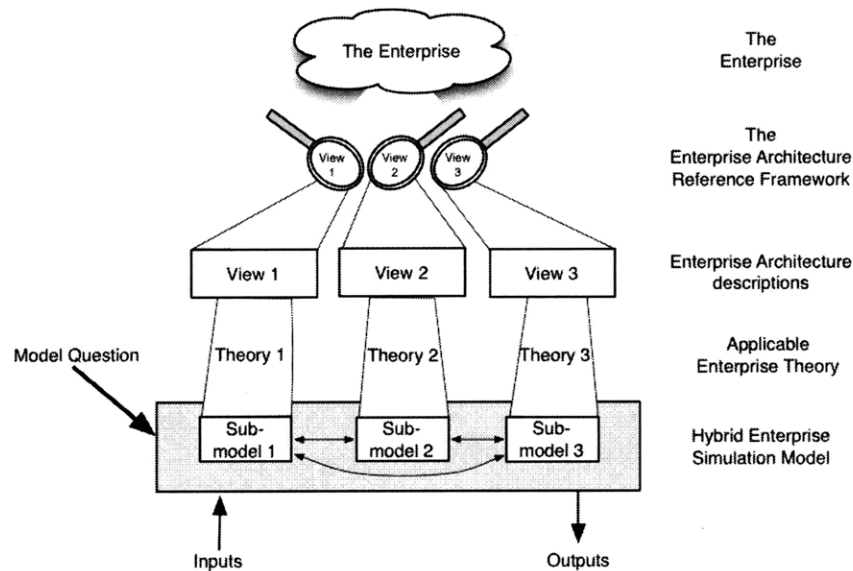


Figure 1-2: The proposed method for creating hybrid, enterprise architecture based simulation models

This thesis proposes an approach outlined in Figure 1-2 for the creation of hybrid, enterprise architecture based simulation models that can be used to analyze enterprise behaviors driven by the enterprise's architecture. At the top of the figure, the complex, real world enterprise is represented as a cloud. An enterprise architecture framework, shown as a set of lenses, is used to focus and abstract the enterprise to produce the enterprise architecture, shown as a set of three boxes representing three possible views within the enterprise architecture corresponding to the framework used. This enterprise architecture is a static representation of the enterprise, and primarily serves a communicative role.

When enterprise leaders have a question about how their architecture may affect the behavior of the enterprise, a model can be constructed using the question and the boundaries and structures identified in the enterprise architecture. Theoretical constructs from pertinent bodies of enterprise theory are then used to guide the creation of the sub-models using a simulation methodology that is matched to the representative behavioral dynamics of each view.

There are many potential benefits of such an approach to simulating enterprises. By modeling multiple perspectives of the enterprise simultaneously, hybrid simulations can be created which do a much better job of analyzing behaviors driven by interactions across these perspectives. Without a hybrid approach to simulation, these cross-enterprise interactions cannot be effectively simulated, and enterprise leaders will not have any tools at their disposal to investigate the effects of architecting choices or to explore high-performing variations to the architecture. Without such an analysis capability, enterprise architecting will remain far more art than science.

Using an enterprise architecture to guide the development of the simulation also has the important benefit of providing a pre-existing abstraction of the enterprise for the modeler to use, eliminating the need to structure the model in an *ad hoc* fashion based upon the modeler's independent investigation. Using this existing abstraction makes model component reuse a possibility, as sub-models will all use the same boundaries and interfaces. This allows an enterprise to build up a library of sub-models that can be more quickly interfaced to create hybrid models to answer new questions. Simulation models based on enterprise architecture also aid communication of the model to stakeholders familiar with the enterprise architecture.

1.5 RESEARCH OBJECTIVES

The hypothesis guiding this work is that hybrid simulation modeling of enterprise structure and behavior, based on the enterprise's current documented architecture and making use of enterprise architecture reference frameworks, can provide useful insights into the effect of the enterprise's existing architecture on its overall performance. This insights developed by this approach are otherwise not possible if the enterprise leadership were to rely on "business as usual" methods, for example using past experience, common sense, or trial-and-error approaches.

This thesis will explore how a process and methodology for creating hybrid simulation models of enterprise behavior using the enterprise's architecture can be created and applied in practice. A process will be developed based on an understanding of the extant literature in enterprise architecting, various enterprise theories, and simulating modeling to create these simulation models specific to an enterprise's architecture. This thesis will investigate properties of an appropriate enterprise architecture framework for this purpose as well as how simulation models can be aligned with them. This research will also review possible methods for testing and evaluating such models for usefulness to the model users and consumers.

This methodology will also be put into practice in a proof-of-concept case study of a real-world enterprise to see if such models can provide insight into enterprise behavior not possible using existing methods. The developed process for created hybrid simulation models will be applied to the dynamics of growing new business in a medium-large scale aerospace enterprise. This model will use multiple views from the enterprise's architecture to build an interconnected, hybrid simulation consisting of discrete event, system dynamics, and agent-based sub-models capable of capturing many of the behaviors that influence the ability of the enterprise to obtain future business. After testing, the hybrid model will then be used to identify critical points of leverage within the architecture to affect enterprise performance, and to architect and compare alternative architectures, demonstrating the utility of the approach.

1.6 THESIS OUTLINE

Chapter 2 of this thesis will begin with an investigation of enterprise complexity, identifying critical ideas and theories that will help to manage and abstract the enterprise. This will follow with a discussion of the literature on enterprise architecting and enterprise architecture frameworks and their role in abstracting the enterprise. This discussion will conclude by examining related enterprise

theory and how it may be brought to bear in understanding and modeling enterprise complexity. Chapter 3 will then review the literature related to enterprise modeling, discussing previous modeling attempts as well as simulation methodologies that hold promise for capturing various aspects of enterprise behavior. The chapter will then highlight some previous work in hybrid modeling, and discuss how these models may interconnected to form the hybrid simulation. Chapter 4 formally develops the proposed methodology of hybrid simulation modeling employing enterprise architecture frameworks, and presents a process for developing and evaluating the models.

Chapter 5 will begin the case study of the “TechSys” aerospace company, by providing background information on the enterprise, its challenges, its architecture, and the objectives of the simulation model. Chapter 6 will then walk through a step-by-step application of the process developed in Chapter 4 as it was applied to creating the TechSys Simulation Model. Chapter 7 presents an analysis of the model and its outputs, and describes the insights and benefits gained from its application to TechSys. Chapter 8 will conclude the thesis with a summary discussion of the efficacy, benefits, and challenges of this methodology, and will present opportunities for continued work to develop it into a viable, effective tool for enterprise management.

Chapter 2: MANAGING ENTERPRISE COMPLEXITY WITH ENTERPRISE ARCHITECTURE FRAMEWORKS AND ORGANIZATIONAL SCIENCE

A manager faces tremendous challenges in the struggle to develop and manage a successful enterprise. A manager will often be forced to make a series of tradeoffs in the design and operation of an enterprise in order to meet the challenges posed by the breakneck pace of change in competitive markets, difficulties aligning the interests of many stakeholders, and conflicting strategic priorities. Unfortunately, there is rarely a clear, equivocal linkage between a change in the foundations of an enterprise's design and its resulting behavior. Unlike a machine that can be described using the laws of physics, an enterprise's behavior over time cannot be predicted using a set of equations derived from the enterprise's structure. Enterprises demonstrate complexity; that is, they are comprised of many components that interact non-deterministically and with feedback in ways that produce system-level emergent behaviors that cannot be easily predicted based on an analysis of their components individually.

Complexity presents serious challenges to managers who seek to understand how the design and alignment of an enterprise's structures, strategies, policies, incentives and processes can enable it to meet its strategic goals. The complex nature of enterprises makes it extremely difficult to untangle cause from effect, as causal mechanisms can often be lost in a web of interactions separated in space and time. A change to a process, for example, can have both its intended effect

as well as unintended side effects in seemingly unrelated parts of the enterprise. Tackling complexity in enterprises requires a *holistic* approach; that is, the enterprise must be studied as a system, rather than reducing it to the collection of its components. Tools must support such an approach and methods based on sound theoretical and practical foundations that make the analysis of complex systems possible.

The goal of this chapter will be to explore the literature related to the study of complexity of enterprises in order to synthesize a framework suitable for guiding the creation of an analysis approach focused on understanding complex enterprise dynamics that can be used to aid enterprise management. The chapter will begin by reviewing the application of complexity theory to the enterprise, highlighting principles such as “systems thinking,” near-decomposability, scale, and perspective and how they can be brought to bear in the analysis of enterprise behavior. Next, the concept of enterprise architecture will be introduced as a holistic construct for managing enterprise complexity, with an eye towards its application for understanding enterprise dynamics. Enterprise architecture is supported by two major knowledge streams: enterprise architecture frameworks, which provide a template for the enterprise architecture, and organizational science, which provides the theoretical underpinnings and practical that make the creation of simulation models possible and useful. This chapter will conclude with a discussion of how both enterprise architecture frameworks and the theories and constructs of organizational science can both be brought to bear in order to create a methodology capable of understanding complex enterprise dynamics.

2.1 COMPLEXITY IN ENTERPRISES

While organizations have long been considered “complex” in the casual use of the term, the study of them as *complex systems* did not begin until the 1950s and

1960s as a result of research done in the fields of General Systems Theory (GST) and cybernetics. These contemporary fields together gave rise to modern notions of systems and complexity, although they studied different aspects of complexity in systems. GST sought to identify and understand common structures, behaviors, and attributes of many different kinds of systems across scientific disciplines, ranging from biology to physics to sociology (von Bertalanffy, 1956). Cybernetics studied the capability of systems for self-regulation through feedback and environmental sensing. Although cybernetics often focused on the study of technical systems, its followers equally applied their analysis to organizations and other socio-technical systems.

The term *complex systems* was used by both GST and cybernetics to describe a general class of systems that exhibited behaviors that were difficult to predict. Herbert Simon, in the seminal work on the architecture of complexity, notes that

“by a ‘complex system’ I mean one made up of a large number of parts that interact in a non-simple way. In such systems, the whole is more than the sum of then parts, not in an ultimate, metaphysical sense, but in the important pragmatic sense that, given the properties of the parts and the laws of their interaction, it is not a trivial matter to infer the properties of the whole.” (Simon, 1962:2)

Complex systems can be technical or social systems, or both; their key feature is the interaction of their parts. Even cybernetics, which tended to focus on technical systems such as fire control radars, saw how analyzing organizations in terms of the interaction of their major elements could help in understanding their behavior. Norbert Weiner, one of the fathers of the cybernetics movement, noted that “we must consider [organizations] as something in which there is an interdependence between the organized parts” (1956:322). If the interdependence of the parts could be understood and characterized, the behavior of the system could be understood.

Beer (1964) classified three major classes of system complexity. The first class is composed of those systems that are both simple and deterministic. A second class exhibits elements of complexity and is probabilistic (e.g., a production line), while the third class, which includes enterprises, is “exceedingly complex” and probabilistic. The second class can be described and *predicted* using established statistical techniques and is considered to be the realm of operations research and systems engineering; the third class has proven much more challenging to analyze, and the efforts to understand these more complex, dynamic systems gave rise to cybernetics, and later to fields such as system dynamics and complex adaptive systems (Scott and Davis, 2007). This third class of exceedingly complex systems, such as enterprises, demands new techniques and perspectives for analysis. No longer can these systems be easily decomposed for study—they require analysis as a whole system.

2.1.1 A “systems approach” and a “reductionist approach” of complex systems

As “exceedingly complex” systems, the study of enterprises requires a holistic approach. That is, an enterprise must be studied as a system, rather than as a collection of its components. A “systems approach” to the study of enterprises stipulates that enterprises cannot be understood solely by an analysis that attempts to decompose them into ever increasing levels of detail. Such a reductionist approach “gives us only a vast number of separate parts or items of information, the results of whose interactions no one can predict. If we take such a system to piece, we find that we cannot reassemble it!” (Ashby, 1956:36)

Complex systems resist reductionist approaches to analysis because interconnections and feedback loops preclude holding some elements constant in order to study others in isolation (Anderson, 1999).

It is non-trivial, however, to analyze the behavior of a complex system using a purely holistic approach; to some degree, decomposition is also required. As Simon notes, “in the face of complexity, an in-principle reductionist may be at the same time a pragmatic holist” (1962:2). In other words, some system decomposition is needed in order to understand the basic functions of the system, but must be done in a way that still permits analysis of the system as a whole, accounting for feedback and dynamics across the decomposition. Any effective approach to the analysis of a complex system such as an enterprise must balance the need for decomposition with an appreciation of the system function as a whole.

2.1.2 Near-Decomposability of Complex Systems

The search for balance between a holistic approach and a reductionist approach to analyzing complex systems can be found in Simon’s work on the architecture of complexity. Simon found that complex systems inherently tend to be hierarchical systems, and those hierarchical systems exhibit principles of what he terms “near-decomposability.” Near-decomposability is the concept that over the short-term, the behavior of a subsystem exhibits approximate independence from other subsystems, and that over the long-term, the behavior of subsystems is dependent on the other subsystems in an aggregate way (Simon, 1969). The hierarchical systems that Simon refers to are not necessarily hierarchies in the notion of a Weberian hierarchical organizational structure where organizational structures are decomposed along rigid, top-down rank-based hierarchies, but instead are systems “composed of interrelated subsystems, each of the latter being, in turn, hierarchic in structure until we reach some lowest level of elementary subsystem.” (Simon, 1969:467) This definition encompasses a broad array of natural, physical or engineering systems, as well as organizational systems embracing both tree-like “pure” vertical hierarchies and lateral hierarchies with horizontal links at various levels.

The concept of near-decomposability in complex systems suggests an architecture composed of highly internally coupled subsystems with looser

external couplings between subsystems. The subsystems of a complex system can be understood and analyzed quasi-independently of each other, and the system as a whole can be understood by analyzing the interactions among the subsystems, without having to uncover detailed micro-level interactions throughout the system. See Figure 2-1. Complex systems that employ properties of near-decomposability have been shown to be more stable, more adaptable, and have greater evolutionary fitness (Simon, 1962, 1969).

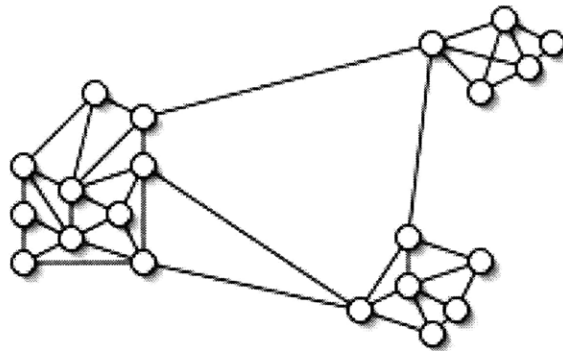


Figure 2-1: An example of the structure of a nearly-decomposable system consisting of highly interconnected subsystems with loose connections between subsystems. The length of the arcs connecting the nodes is representative of the frequency of interaction—the shorter the arc, the more frequently the two nodes communicate.

A second observation made by Simon is that not all elements in a system are tightly connected in a causal fashion. Some exhibit “loose coupling”: causal connections between components that are weakly connected, allowing for more autonomy between elements or subsystems. When two elements are loosely coupled, a “dependent” element is influenced rather than controlled. It may be more likely to change in reaction to a change in another, independent, element, but in reality does not necessarily change. The literature considers organizations to be loosely coupled, such as where rules do not always govern behavior (Scott and Davis, 2006). This can be seen in an enterprise when a new policy or rule change is made and the involved people do not change their behavior in

accordance with the policy change. Although such changes do influence behavior, they do not determine behavior. This linkage is an example of loose coupling in a complex system.

The property of near-decomposability has many important implications. Recognizing and understanding subsystems within a complex system as well as the vertical and lateral lines of interaction between subsystems reduces the apparent complexity of the system, making analysis of the system more tractable. In complex systems, every component is not necessarily connected to every other component; components are connected to other components to form sub-systems; the sub-systems in turn are connected to form the system. By being able to understand the system at the level of its subsystems, analysis of the system's behavior is made much more manageable. Instead of studying all possible interactions within a complex system, analysis can focus on the interactions between subsystems, assuming that the behavior of the subsystems can be understood.

2.1.3 Scale in Complex Systems

Subsystems can be further decomposed into components if their own complexity makes this necessary. In this way, a notion of scale (degree of resolution) in complex systems is introduced, where the behavior of any given level in a system can be determined by the interaction of elements at the next lower scale (Bar-Yam, 2003). For example, the behavior of a corporate organization can be studied at multiple scales. At the highest "system" level, the behavior of the corporation can be understood by studying the interaction among its operating units, how the respective operating units affect the overall system behavior, and how the overall system, in turn, may affect the behavior of the individual operating units. Operating units, however, display their own complexity and may require further decomposition into further instantiated business units. The behavior of each operating unit can be understood by studying behavioral

dynamics at the next lower scale, that of its specific business or product lines. This scale can be further reduced, down to the atomic level of individuals if necessary, in order to determine the causal structure of behaviors at higher scales. The power of multi-scale, nearly decomposable, systems is that they rarely need to be decomposed to an atomic level: the behavior of the system can often be understood reasonably well in terms of the interactions at the higher-level scales.

2.1.4 Perspective in Complex Systems: “top-down” versus “bottom-up”

The early work that comprised the study of complex systems, such as cybernetics and later system dynamics, focused on understanding how the interrelationships among stable, causal structures in complex systems drove system behaviors (Sterman, 2000). In such cases, relatively simple feedback and control structures in a system give rise to fairly complex behavior. These macro-level features serve as the basis for analysis and the creation of descriptive models of systems. This perspective of complex systems takes what is termed a “top-down” perspective; that is, a system’s behavior can be understood and managed by looking at the interaction of subsystems at higher scales, and changes at the top will cause lower levels to work accordingly.

In top-down analysis and design, finer scales are often aggregated and simplified in subsystems. Subsystems are sometimes modeled as stochastic processes, without regard to the underlying micro-level mechanisms giving rise to the macro-level behavior. Top-down analysis emphasizes how the pathways of communication and the structure of the system drive its behavior at a macro-level that can often be modeled using differential equations such as those employed by cybernetics (Weiner, 1956) and system dynamics (Forrester, 1962).

The alternative to a “top-down” perspective of complex systems is the “bottom-up” perspective, which seeks to understand how the micro-level interactions and

incentives within a complex system in turn drive macro-level behaviors (Schelling, 1978). These micro-to-macro behaviors are known as “emergence,” or “emergent behaviors.” The study of how the micro-motives of individuals within a social system drive the macro-behavioral phenomena has been a long-standing issue in the study of sociological behaviorism, exchange theory, rational choice theory, and economics (Sawyer, 2003) as well as organizational science and design (Anderson, 1999). Agent based models, the popular tool of choice for analysis of bottom-up behaviors, has allowed researchers to investigate a wide range of bottom-up behaviors, from the phenomenon of standing ovations (Miller, 2004) and civil violence (Epstein, 2002) to evolution (Sigmund, 1993).

The stock market is one example of a complex system whose behavior is driven strongly by bottom-up forces and exhibits emergent behavior. The overall behavior of the stock market is the result of millions of investors making local or micro-level buy-sell decisions that aggregate through multiple levels into national or global trends. There is no strong, central, directed governing structure that controls a stock market and tells it how to perform its task. Likewise, such a process could not be described using an aggregate set of differential equations describing relationships with feedback. Each atomic element in the system, the individual investors, makes his or her own decisions according to his or her own rules, risk preferences and incentives. The “design” of a stock market is the design of the rules that govern the actions that investors can or cannot take. They make locally beneficial decisions in response to these rules, giving rise to bubbles, crashes, and recessions.

Table 2-1: Table comparing characteristics associated with top-down and bottom-up approaches to systems analysis

Top-down	Bottom-up
<ul style="list-style-type: none"> • Prescriptive (Directive) • Centralized control • Intended behaviors • Focus on higher scales • Stable environments 	<ul style="list-style-type: none"> • Descriptive (Information-giving) • Decentralized Control • Emergent behaviors • Focus on the lowest scale • Changing environments

Both top-down and bottom-up approaches have found followings in the study of complex systems. Each offers a different perspective and insight into system behavior, as highlighted in Table 2-1. Top-down analysis allows for the study of the design of central control mechanisms, management, processes and feedback. Bottom-up analysis allows a better understanding of emergent behaviors, often unintended, arising from the interaction of elements in the system. Designed, goal-oriented complex systems such as enterprises generally require both analytical perspectives: the bottom-up perspective is crucial to understanding how individuals are incentivized to perform (in an intended or unintended fashion), while a top-down perspective is needed to provide the structure and processes necessary to coordinate all of their activities in a desired direction. Few real world complex systems involving people can be completely described purely from a top-down or a bottom-up perspective; most have some features that lend themselves to one perspective, and some features that lend themselves to the other.

2.1.5 Applying Complexity Theory to Enterprises

The study of complex systems requires a balance of contrasting perspectives depending on the context of the system: holism versus reductionism; top-down versus bottom-up. The perspectives and analytical approaches taken in a study of a complex system is contingent on the system under study and the particular

dynamics being investigated. There is no single approach that is best suited to the study of all complex systems and for all questions (Mingers and Gill, 1997).

The study of enterprises as complex systems presents researchers and managers alike with a broad array of challenges. Enterprises are both designed from the top down and exhibit emergent behaviors from the bottom up; they are both centrally directed and dependent on the actions of agents who make locally rational decisions dependant on their local incentives. Accordingly, different bodies of research and practice have been built around different approaches for managing the complexity of the enterprise.

The next sections will review two major areas of study that approach enterprises as complex systems: *organizational science* and *enterprise architecture*. The organizational science literature tends to focus on theory generation and generalizable observations regarding the structure of organizations, especially in regard to the structure of the organization in response to its environment. In contrast to organization science, enterprise architecture takes a more top-down approach to analysis of enterprises as complex systems. The field of organization design takes a holistic, management-focused approach to the design of the enterprise at a high, but interconnected scale; the traditional practice of enterprise architecting, in contrast, takes top-down reductionist approach to describe the complexity of the enterprise for purposes of system design and implementation.

These fields employ a variety of approaches that have been used to break through the barriers posed by complexity and gain a deeper understanding of the drivers of enterprise behavior. By reviewing each field and studying their relative strengths and weaknesses, we can begin to establish the basis for the creation of a hybrid approach that captures the best elements of each of the approaches, and mitigates their shortcomings.

2.2 ORGANIZATIONAL SCIENCE

At its core, organizational science is the study of the structure, behavioral dynamics and design of organizations. It seeks to identify scientific principles that can be employed to describe how organizations are structured and how they behave given their environments, and offers suggestions on how new organizations can be designed. Organizational science uses the organization as its unit of analysis, which is bounded by organizational boundaries and functions. This can be seen in contrast to the study of enterprises, which is often more practically oriented and uses the enterprise as its unit of analysis, which extends the organizational construct to incorporate an extended value chain, its stakeholders and its environment.⁴

The modern study of organizational science emerged after the Second World War, when sociologists and engineers alike turned their attention to the study of organizations, bringing with them their unique perspectives. Scott and Davis (2006) identify two important strands that came together to form the foundations for organization studies: a “rational” perspective to the study and design of technical and administrative systems, and a “natural,” or humanist, perspective from social psychologists and sociologists who emphasized the human and social features of organizations. The rational perspective, descended from the Scientific Management movement in the early 20th century (Taylor, 1911) and the later work of Fayol (1949), tended to be highly analytical of processes and structures within organizations with little regard for non-quantifiable aspects of the organization; the natural perspective placed an emphasis on understanding decision making and choice within organizations from the perspective of individuals and social groups.

⁴ This observation is a broad generalization with many exceptions. In practice, there are many organizational researchers who research topics well outside core organizational issues. While the use of the term “organization” can vary in intended scope, the term “enterprise” is almost always used to indicate a broader perspective and scope. This section will use the term organization rather than enterprise for consistency with the literature.

A third approach, termed the “open system” perspective, emerged as the result of the application of systems thinking to organizations in the 1960s: specifically, viewing organizations as open systems that are driven by internal and external interactions. Both general systems theory and cybernetics heavily influenced the “open systems” perspective, with the stipulation that organizations typically display a much greater degree of loose coupling than most systems studied by cybernetics. This perspective of organizational science espoused theories based on locally rational but cognitively limited actors loosely coupled within the organization (March and Simon, 1958). The Open System perspective within organizational science contains some of the most valuable theories from the perspective of an enterprise architect.

2.2.1 The Open System Perspective of Organizations

While both the “rational” and the “natural” perspectives of organizational science have contributed greatly to the understanding of organizations, the “open systems” perspective is more suited to understanding the broader concept of *enterprise*. The open systems perspective places an emphasis on challenging the boundaries of the organization while also examining the nature of its interdependencies (Thompson, 1967). It also recognizes that organizations often exhibit a high degree of loose coupling (Cyert and March, 1963), which can allow the system to be more highly adaptive and flexible (Orton and Weick, 1990). This perspective is most likely to include analysis of an enterprise’s environment, processes, technologies and products in addition to organizational forms and decision-making (Scott and Davis, 2006). The open systems perspective tends to take a more holistic approach to analysis than either of the other two perspectives of organizational science. From the vantage point of the enterprise architect, the primary areas of interest within the open systems perspective of organizational science are contingency theory and the closely related field of organizational design.

2.2.2 Contingency Theory

The most well known school within the opens systems perspective of organizational science is contingency theory. Contingency theory holds that the many factors that shape the design of an organization should be chosen contingent on the environment of the organization, and seeks to identify these factors and the environment for which they are appropriate. In establishing contingency theory as applied to organizational design, Lawrence and Lorsch (1967) proposed that organizations are aligned to their environment on at least two levels: 1) the structural features of the organization should relate to the environment, and 2) the approach to differentiation and integration within the organization should be consistent with the overall complexity of the environment. Since this initial work, many others have expanded contingency theory by identifying other factors on which organizational design should be contingent, including size, scale, strategy, technology, geography, risk management, resource dependency, cultural differences, scope, and organizational lifecycle (Lawrence, 1993). The mantra of contingency theory might be captured by Galbraith, who says "There is no one best way to organize; however, any way of organizing is not equally effective."(1973:2)

Contingency theory as a field seeks to build an ever-increasing compilation of factors upon which the design of the organization is contingent. Burton and Obel (1995) provide a fairly comprehensive collection of contingency theory observations, and went on to develop software to help managers make use of contingency theory by identifying relevant theories given a situation. These contingency theories, identified through fieldwork and statistical analysis of real-world data, are used to develop prescriptive guidance for the design of organizations, such as "high technology development environments should not have high organizational complexity" (Caroll, et al., 2006). This guidance is generalized based on past observation, and it may not necessarily hold as business environments adapt and change. Contingency theory provides no means of analysis of an existing organization, except so far as elements of an

organization's design can be checked against the collected body of literature to identify any potential mismatches. While contingency theory can not be used as a basis for the construction of simulation models of an enterprise's behavior, it is still extremely useful for providing guidance when initially designing an enterprise or when adapting and enterprise in the face of changing environments, and is the dominant approach to organizational design today (Lawrence, 1993).

2.2.3 Organizational Design

The field of organizational design soon followed, capitalizing on the work of contingency theorists with the aim to provide prescriptive guidance to those wishing to design⁵ organizations. In addition to the study of organizational contingencies, proponents of organizational design advocated the study of such areas as work flows, control systems, information processing, planning mechanisms, knowledge transfer and their interactions. The field of organizational design school tends to be more applied than most other fields within organizational science; organizational designers tend to seek change and improve organizations from a managerial perspective, rather than describe and understand them from a theoretical perspective (Scott and Davis, 2006).

An organizational designer embraces the complex systems philosophy of “the importance of treating the system as a system—as more than the sum of its component elements” (Scott and Davis, 2006: 99). As such, heavily emphasis was placed on understanding the fit and interdependency of components of an organization's design. Thompson, a contingency theorist, is considered an early pioneer of organizational design for his work identifying and categorizing interdependencies in organizations and the dimensions of coordination in organizations (Thompson, 1967).

⁵ The verb “design” is used equivalently to the verb “architect” in the case of organizational design.

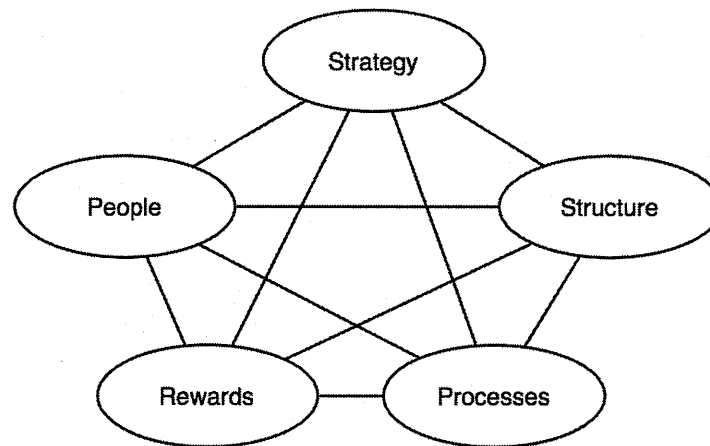


Figure 2-2: Galbraith's Star Model (1973)

Jay Galbraith was one of the earliest organizational designers to clearly separate themselves from contingency theory. His work on the design of complex organizations sought to establish a normative approach to organizational design by establishing a model for creating an organization from the ground up based on the information processing needs of the organization (Galbraith, 1973). Galbraith developed what he termed a "star model" as a framework for designing organizations (See Figure 2-2). The star model consists of five categories that all design policies will fall into: strategy, people, structure, rewards, and processes. All of these categories can influence each other, and must be considered as a system, driven by strategy. After strategic policies have been developed, Galbraith holds that design policies for the structure, processes, people, and then rewards should be developed, in that order (Galbraith, 2002). In taking an information processing view of organizations, Galbraith further identified how boundedly rational actors within an organization could be overloaded with information, and established how their relative ability to process information could impact organizational performance (1973). While much of his work the information processing perspective of the organization has been widely cited and used, Galbraith's star model has been critiqued by other organizational scientists for being overly qualitative and lacking a means of validation (Levitt, 2004).

Organizational design, true to its roots in systems thinking, advocates that the organization must be understood as a holistic system composed of subsystems that must be kept in balance to some extent. Nadler and Tushman(1980) hypothesized that “all things being equal, the greater the degree of congruence, or fit, between the various elements, the more effective the organization will be.” They went on to establish their own model of organization design, based upon four “domains”⁶ that they identified as being key to organizational design: (1) the work to be done, (2) the individuals involved, (3) the formal structures, and (4) the informal structures (Nadler and Tushman, 1997).

These normative models for designing organizations presented by Galbraith (1973) and Nadler and Tushman (1997), among others, share many similarities with enterprise architecture frameworks, with a few key distinctions. The scope of the organizational design models is typically narrower than enterprise architecture frameworks, as they are limited to the boundaries of classic organization (although Galbraith’s model is perhaps slightly broader, as it also includes processes). The focus of these models on the organizational aspects of the enterprise provides a contrast to enterprise architecture frameworks, which tend to be much broader in scope while maintaining an emphasis toward the information and technology needs of the enterprise. One critique of both enterprise architecture frameworks and organizational design models is that the partitioning approach used by both fields cannot be tied back to any empirically derived theories for decomposing the enterprise (Levitt, 2004). The frameworks and models are intended to be useful, practical tools, for creating and managing a wide array of enterprises, and not to be rigorously derived taxonomies.

The categories, domains, or views employed by organizational design models and enterprise architecting frameworks each decompose the enterprise in a

⁶ There is no consistency within the literature for the term used to describe the groupings in organizational design models. The terms “categories” and “domains,” appear to be used interchangeably.

manner that is well suited to particular analytic approaches. The domains used by Nadler and Tushman (i.e., formal structure, informal structure, nature of work, and individuals) are useful for a social analysis of the enterprise, while Galbraith's categories are focused on a broader alignment of the organization with incentives and processes to achieve operational goals, without as much focus on a social analysis of the organization. Enterprise architecture frameworks, as shown in Section 2.3.2, are much more focused on the implementation and integration of the information systems required to support business processes, with little emphasis placed on understanding social aspects of the organization. None of these models are necessarily wrong, but rather are tools suited for different tasks. At the end of the day, "essentially, all models are wrong, but some are useful" (Box, 1987:424).

Not all research within organizational design has focused on the development of models for designing organizations, however. Much research has also been conducted to identify generic constructs and patterns that could be useful to the designers of organizations. Mintzberg (1983), for example, sought to establish a set of organizational constructs that could be used to understand generic organizational structures. The six generic organizational constructs identified by Mintzberg are:

- *operating core*: the people directly related to the production of services or products;
- *strategic apex*: serves the needs of those people who control the organization;
- *middle line*: the managers who connect the strategic apex with the operating core;
- *technostructure*: the analysts who design, plan, change or train the operating core;
- *support staff*: the specialists who provide support to the organization outside of the operating core's activities;
- *ideology*: the traditions and beliefs that make the organization unique.

Each of these areas is defined in terms of the roles played within an organization. The basic role of the generic organization is to divide labor into distinct tasks, and then coordinate across the areas to complete the tasks. Mintzberg argues that as organizations emphasize one of these basic areas, different generic organizational structures result. The five generic structures are

1. *simple structure*, which emphasizes the strategic apex;
2. *machine bureaucracy*, which emphasizes the technostructure;
3. *professional bureaucracy*, which emphasizes the operating core;
4. *divisionalized form*, which emphasizes the middle line; and the
5. *adhocracy*, which emphasizes the support staff.

Each of these generic structures is best suited for particular business environments. By identifying an organization's operating environment, an organizational designer/architect can use Mintzberg's constructs to help craft an organizational structure that is well suited for that environment.

The most recent addition to the field of organizational design is the dynamic capabilities literature, which seeks to establish, categorize, and analyze the links between an organization's "dynamic capabilities" (its ability to sense, seize, and manage threats from the environment) and its "microfoundations"—the organization's skills, processes, procedures, organizational structures, decision rules, and disciplines, i.e., its architecture. This literature strives to explain the "sources of enterprise-level competitive advantage over time, and provide guidance to managers for avoiding the zero profit condition that results when homogeneous firms compete in perfectly competitive markets" (Teece, 2007). It establishes two distinct dynamic capabilities that organizations must be aware of: their "technical fitness" and their "evolutionary fitness" (Helfat et al., 2007). Technical fitness is a measure of how well an organization's capabilities performs their functions; evolutionary fitness is a measure of how well its capabilities

enable it to make a living within its environment. A survey of this growing literature can be found in Helfat et al.(2007).

In many ways, the dynamic capabilities literature seeks to bridge the gaps that exist between organizational design and contingency theory by seeking to establish normative theory and frameworks that are useful to managers seeking to shape and guide their enterprise's behaviors, and perhaps is one of the most useful. Teece's Dynamic Capabilities Framework is the most recent and complete framework from this literature to establish a link between a firm's architecture and its dynamic capabilities. See Figure 2-3 for a representation of the framework.

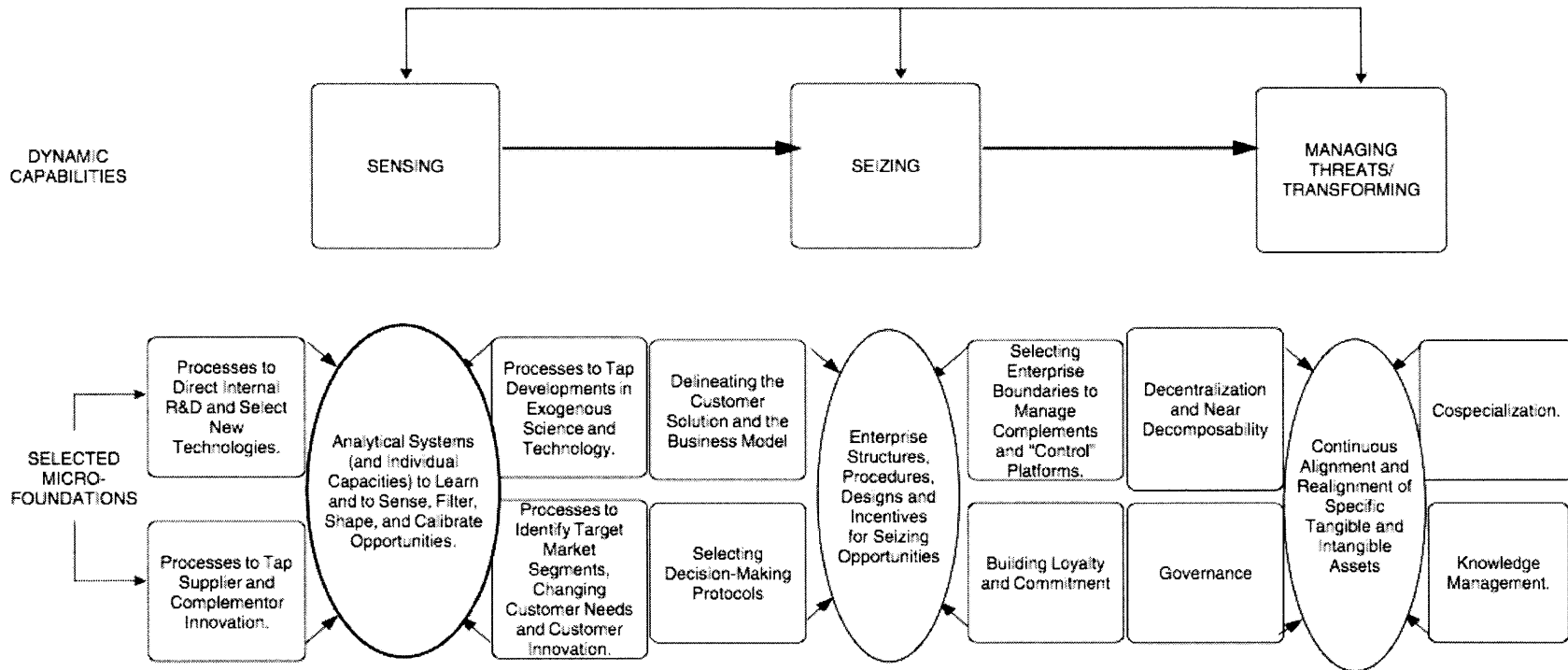


Figure 2-3: Teece's Dynamic Capabilities Framework (2007)

Teece's framework divides the microfoundations of the architecture into three primary groups, aligning them with either the enterprise's ability to sense and respond to the environment, seize opportunities, or manage threats, as shown in Figure 2-3. It has a much broader scope than previous frameworks from organizational design, encompassing aspects of the architecture including knowledge management, governance, multiple processes, decision-making tools, and incentives. While the Dynamic Capabilities Framework is a very useful taxonomy for those designing and managing an enterprise, it falls short of its full potential. Like many other frameworks, it does not take into account interactions among the microfoundations of the architecture. The framework itself is fairly hierarchical, which runs the risk structuring the architecture in a siloed fashion. Additionally, although the framework seeks to align the enterprise architecture with its dynamic capabilities, it provides no tools or methods for ensuring that this alignment is achieved. Similar to much of the contingency theory literature, this framework is intended to be used to determine the adequacy of fit in an organization's design, rather than to provide the basis for a further quantitative analysis of to what extent the enterprise's "microfoundations" support its dynamic capabilities. The ability to perform quantitative analysis of an enterprise's architecture is a key capability that would provide a great value to managers.

2.2.4 Computational Organizational Science

Quantitative analysis of the architecture of enterprises is largely missing from both the contingency theory and organizational design literatures. There are few tools to quantitatively assess tradeoffs made in organizational design, or to assess one architecture against another. Due to the inherent complexity of enterprises, closed-form mathematical or logical analysis of designs is limited to only the absolute simplest structures, such as the analysis of the complementarity of structure and strategic fit performed by Milgrom and Roberts(1995). Analysis of modest real-world organizational forms, processes and behaviors requires the use of computer simulation models capable of

modeling the enterprise as a network of interacting, adaptive components over time.

Interest in using simulation models to research organizations is not a recent development. Going back several decades, organizational design theorists advocated the use of simulation models where “all the variables and relationships of interest are linked as understood in a model and then the manager-analyst-researcher manipulates certain ones and observes how others change as the simulation of the system plays out”(Swinth, 1974: 11). Despite the desire to use simulation models, it was many years before both simulation techniques and computational power matured enough to be applied to the types of problems of interest to organizational scientists.

Carley(1995) was among the first to advocate the creation of a new discipline within organizational science, to be known as “computational organizational science,” arguing that virtual experiments employing simulation models could be used for the purpose of creating and testing organizational theory, especially for the application of complexity theory to the organization to understand how they adapt and evolve. Simulation models allow organizational researchers to create controlled, easily manipulated experiments where real world “natural experiments” inside organizations are not feasible due to cost, scope, or ethical considerations (Fowler, 2003). Techniques such as agent based modeling⁷ allow the creation of simulation models that capture the behavior of an organization at multiple scales by simulating the interaction of its elements. Such models allow analyses of how organizations search, adapt and learn in changing environments.

Computational organizational science has come to focus primarily on taking a complex *adaptive* systems view of organizations, and has used simulation models to establish theories for how organizations evolve and adapt in uncertain

⁷ Agent based models are discussed in further detail in the next chapter

environments, especially from a bottom up perspective (Anderson, 1999). Although still in its infancy, computational organizational science shows much promise at developing theory that helps researchers better understand how enterprises adapt at multiple scales to their changing environments.

Computational organizational science primarily uses simulation models in a theory-generative or theory-testing role, rather than as an analysis tool in the process of organizational design. The models employed to date been highly abstract “toy” models that served to prove a hypothesized mechanism, rather than as a simulation of a real-world organization or system. While such simple models have proved valuable in the creation of new organizational theories governing the evolution of organizational design(see Rivkin and Siggelkow, 2003; Ethiraj and Levinthal, 2004), they cannot be considered tools that can be used by managers as an analysis tool in organizational design. These “toy” models are far too abstract for managerial application. The aim of computational organizational design is to develop new theory, not to create new architecting and analysis tools.

Breaking from the tradition of using simulation modeling exclusively for theory generation, Levitt (2004) argues that the development of analysis tools for organizational design has been hampered by the lack of robust theory that supports it. He then argues that this is changing, and that two major strands of organization research have provided the foundations for a theory strong enough to support analysis tools that can be used to design organizations:

1. The information processing view of the organization (March and Simon, 1958; Galbraith, 1973) provides the framework for agent-based simulation models at the micro levels through an understanding of information pathways and local incentives.
2. Contingency theory provides the macro-level propositions that relate structural form to context, based on empirical observation.

By bringing together theory of organizations from both top down and bottom up perspectives, Levitt argues that simulation models can link together these theoretical perspectives and should be used to analyze the potential performance of organizational design in different environments. He states that simulation technologies have come of age, making the simulation of organizational design both technically possible and theoretically supportable.

2.2.5 Gaps in organizational theory for effective organizational design

Organizational science as a whole has strived to create rigorously developed, generalizable theory with broad explanatory powers. Organizational theory is capable of providing some normative guidance to managers seeking to guide the development of their enterprise, but it does not provide enterprise managers and architects with the tools needed to quantitatively analyze and apply the theory to their own enterprises. While ideally managers would adapt and change their organization's design based upon a careful consideration of current organizational theory, studies have repeatedly shown that managers tend to adapt their organization's design using a costly "trial and error" process based upon prior experience rather than rely on theory (Tatum, 1983). Levitt argues that without the ability for managers and architects to analyze potential changes to their own enterprise's architecture before they are implemented, it is impossible to create an effective architecting process. Without an effective process, the enterprise will revert to using the ad hoc "trial and error" approach (2004).

Additionally, the organizational science literature is highly balkanized into narrowly scoped, independent threads of research. There is little work in organizational science that ties these threads together in any coherent and practical way for architecting in real-world, complex environments. The organizational science literature has been developed in a stovepiped manner,

and as a result, there is little that integrates the disparate areas of research in a coherent manner. This is especially evident in contingency theory, which exists as a collection of often-independent observations of organizational structuring. Teece's work in dynamic capabilities is perhaps one of the best examples in the literature of an integrative framework for relating various facets of organizational theory together to better understand and enterprise's behavioral characteristics, but it falls short of tying together the microfoundations of the enterprise that it identifies, and provides no tools for analyzing the dynamic capabilities of the enterprise.

Organizational science is missing frameworks and theories that can tie together the richness and rigor of its research in a coordinated manner that makes the full body of knowledge more useful to the enterprise leaders managing the development of their enterprises. Until such time as a unifying "enterprise science" can mature and develop, enterprise leaders must look elsewhere to find useful guidance for stitching together stovepiped bodies of knowledge in order to create tools and models to help them better understand and manage enterprise behaviors.

2.3 ENTERPRISE ARCHITECTING

The practitioner-oriented field of enterprise architecting offers an alternative, holistic approach to managing an enterprise's complexity. Enterprise architecting holds that enterprises, as complex systems, can be "architected" in an organized fashion that will give better overall results than an ad hoc or piecemeal approach to organization and design (Bernus, 2003). To "architect" an enterprise is to create and document a specific abstraction of it that describes its fundamental

organization, either in its current state or as a desired future state. This abstraction is known as the enterprise architecture⁸.

The ANSI/IEEE definition of enterprise architecture states that it is “the fundamental organization of a system, embodied in its components, their relationship to each other and the environment, and the principles governing its design and evolution” (ANSI/IEEE 1471:2000). A second, more complete definition is given by Bozdogan, who states that “enterprise architecture is an abstract representation of a ‘real-life’ enterprise’s holistic design (gestalt, configuration, pattern) binding together its structure, strategy, environment and performance, showing its essential elements and the relationships among them, and mapping the interaction between the enterprise and its external environment, as both co-evolve over time” (2007). The enterprise architecture is an abstraction of the “real-world” enterprise structured in such a way that makes both analysis and design possible by treating the enterprise as a hierarchical, nearly-decomposable system. The enterprise architecture has the potential to make the systematic design and improvement of complex enterprises more feasible by describing the enterprise from multiple perspectives and specifying the interactions of disparate aspects of the enterprise.

While the original *raison d'être* for enterprise architecting was to integrate disparate information systems across the enterprise in order to provide value to the enterprise, its application has evolved from enterprise information system integration toward much broader uses. At its core, the purpose of enterprise architecting is to (1) understand the enterprise as an interacting, complex system, (2) communicate the structure and behavior to enterprise stakeholders, and (3) serve as a vehicle for changing the enterprise. It does this not so much by replicating the work done in other fields to understand key aspects of enterprises,

⁸ Many also hold that all enterprises can be said to have an enterprise architecture regardless of whether or not one was explicitly developed, as all enterprises can be described in terms consistent with an enterprise architecture.

but by harnessing them in a coordinated way: enterprise architecting “facilitate[s] the unification of methods of several disciplines used in the change process, such as methods of industrial engineering, management science, control engineering, communication and information technology, i.e., to allow their combined use, as opposed to segregated application” (GERAM v. 1.6.3, 1999). The documented enterprise architecture can be used to foster a shared sense of understanding of the enterprise’s structure, function, and behaviors by enterprise stakeholders, and serve as the blueprint for changing the enterprise. When properly executed, enterprise architecting is a holistic, integrated approach to the design of enterprises based on principles of near-decomposability and the facilitated application of many previously independent disciplines.

Despite its widespread application in industry and government, enterprise architecting has not emerged as a prominent field of academic study. The field has largely emerged from the practitioner literature rather than the academic literature, and as such, it has traditionally not had strong ties to rigorous theory, despite its success and utility in practice. It does, however, offer one of the few practical approaches to integrating a vast array of knowledge about enterprises together with a holistic analytical approach that is able to address the fundamental complexity of enterprises.

2.3.1 Development of Enterprise Architecture Frameworks

Enterprise architecting was born in the late 1980s, emerging in response to repeated failures to integrate information technology and computer-aided manufacturing into existing enterprise operations (Zachman, 1987)⁹. Information technology leaders saw that these system implementation failures were due to the fact that these information systems’ impact reached across the enterprise, and that without an integrated, holistic understanding of the enterprise, it was

⁹ The term “enterprise architecting” was not in common usage until the mid 2000s, although the term has been applied to earlier work developing enterprise architectures.

difficult to successfully integrate such systems into the enterprise. While there had been previous efforts to model the integration of information systems in the enterprise (such as the IDEF family of modeling languages), enterprise architecting differed because focus was placed on understanding the broader enterprise from multiple, integrated perspectives, rather than solely focus on the technical information system and its immediate boundaries. This system-level description of the enterprise was achieved through the application of an enterprise architecture framework, which is a “toolkit” for enterprise architectures that consists of two parts: (1) an ontology for describing the elements and relationships in the enterprise and (2) reference architecture that serves as a guide for creating generic enterprise architectures by identifying key boundaries and interactions. The first such enterprise architecture framework, the Zachman Framework, was released by IBM’s John Zachman in 1987, and has continued to be developed and remains a popular framework in use today.

Throughout the 1990s, the focus on enterprise architecting remained centered on the integration of information technology and systems into the greater enterprise, rather than on the management of the enterprise as a system. Major research initiatives and frameworks of the time, such as the Perdue Consortium’s Purdue Enterprise Reference Architecture (PERA) (Williams, 1998), the ESPRIT Consortium’s Computer Integrated Manufacturing Open Software Architecture (CIMOSA) (CIMOSA, 1996) and the and the Toronto Ontology for Virtual Enterprises Project (TOVE) (Fox, 1992) took an IT-centered approach to enterprise architecting, focusing on how disparate information systems could be created to seamlessly interact and integrate each other. CIMOSA extended this by advocating a shift from activity or functional depictions of system business process centered views (Vernadat, 2001). Little emphasis was given to the “softer” aspects of an enterprise’s design, such as knowledge management (outside of IT considerations) and organizational incentives, product

architecture,¹⁰ or even strategy and the external environment. This approach to enterprise architecting has been more accurately termed “enterprise IT architecting,” to distinguish it from approaches that take an approach that more evenly balances all of the necessary components of an enterprises’ design (Armour, et al., 1999).

Since the turn of the 21st century, a greater emphasis has been placed on enterprise architecting from a position that is more balanced towards viewing the enterprise as an open, strategically-guided system, rather than viewing it from the perspective of IT integration¹¹. This change has gradually come as enterprise leaders and managers have come to see the potential value of using enterprise architecting as a valuable practice beyond its capabilities for the integration of information systems. While enterprise architecting was once an approach that was seen as a methodology of information architects and delegated individuals reporting to the enterprise’s Chief Information Officer, it is increasingly seen as a valuable activity with wide applicability that should be pushed up higher in the organization hierarchy to the office of the Chief Executive Officer¹². More recent enterprise architecture frameworks, such as The Open Group’s Architecture Framework (TOGAF), the Federal Enterprise Architecture Framework, and the Nightingale and Rhodes Enterprise Architecture

¹⁰ In this usage, product architecture refers to the design and structure of an enterprise’s products. This is important, for example, because the design of the organizational and communication structure has been shown to heavily influence the design of products (Conway, 1968).

¹¹ It can be argued that many of the earlier frameworks, such as the Zachman Framework, advocated a more balanced approach to analysis and design of enterprise architectures, but were handicapped by placing enterprise architecting efforts under the domain of the Chief Information Officer.

¹² At a roundtable discussion of Chief Information Officers in the aerospace industry sponsored by the Lean Aerospace Initiative in January 2007, the broad consensus among those in attendance was that enterprise architecting efforts will be more successful as they are pushed out of the CIO’s office and into the CEO’s office.

Framework offer frameworks that are more balanced between the needs of the information technology community and the business community, who are concerned with strategy, organization, processes, and products in addition to information systems. As this trend continues, enterprise architecting will play a greater role in developing future organizational designs.

Enterprise architecture frameworks serve as virtual “toolboxes” for enterprise architecting. They contain tools and methods for enterprise architecting such as architecting processes, modeling approaches, and taxonomies as well as their own reference architecture. Reference architectures are generic enterprise decompositions that serve as templates for the creation of a specific enterprise architecture. Each enterprise architecture framework has its own reference architecture which can be used to decompose an enterprise in a specific way with an intended purpose or focus, ranging from implementation of IT systems (Zachman, TOGAF), enterprise integration (CIMOSA) to capital asset allocation (FEAF). The decomposition of each reference architecture is determined based upon the combined observations and experience of the governing bodies creating them, and has been standardized through the International Standards Organization (ISO).

The Generalized Enterprise Reference Architecture (GERA), part of the ISO 15704 (2000) for the creation of enterprise architecture frameworks, attempts to provide a comprehensive reference architecture that can be used to architect any enterprise for a wide variety of purposes. As the ISO standard, other enterprise architectures could be labeled as “ISO or GERA Compliant” if they fit the structure outlined in the standard. GERA strives to provide a fairly comprehensive decomposition of the function and systems of enterprises with the expectation that a “GERA-compliant” architecture frameworks could employ even finer resolution by further decomposing the enterprise, or choose to use a subset of the decompositions it provides.

2.3.2 Enterprise Architecture Framework Views

Reference architectures provide a template for how enterprises can be decomposed in ways that make their complexity more manageable. The GERA establishes the use of *views* to accomplish the decomposition¹³. “Views contain a subset of facts present ... allowing the user to concentrate on relevant questions that the respective stakeholders may wish to consider. Different views may be made available highlighting certain aspects of the model and hiding all others.”(ISO 15704, 2000). A view in an enterprise framework is an abstraction that defines a perspective of an aspect of an enterprise, such as its organizational structure, its processes, or its information systems. The use of views in enterprise architecture is analogous to the use of views in traditional building architecture—an architect will create a collection of drawings of a building from multiple perspectives to provide stakeholders ranging from owners to contractors a model of the building to be built. These perspectives may vary based on the location of an external observer (floor plan, street view, ¾ view), or based on a functional set of perspective (structural views, electrical views, plumbing views, environmental system views). Without any one of these views, not enough information is present to build the building. It takes the complete set. Unlike traditional architecting of buildings, however, enterprise architecting has no universally agreed upon set of views that can completely specify an enterprise, or even how many views are required to describe it completely. An enterprise is considerably more complex than a building.

¹³ Although ISO 15704 specifies the use of the term “view,” it is not universally used by all frameworks. Some frameworks use “viewpoints,” “perspective,” or “domain.” Although each of these terms has its own particular definition, they are very similar concepts and date back to the Zachman Framework’s use of domains and views.

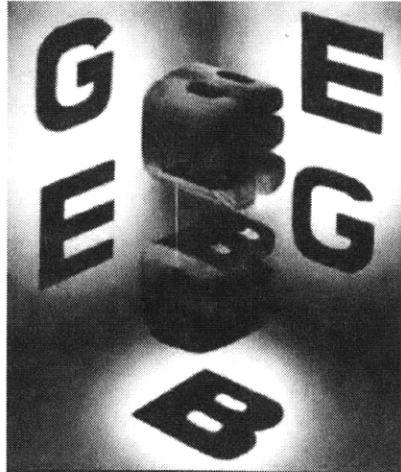


Figure 2-4: The cover image from Douglas Hofstadter's "Gödel, Escher, Bach" (Second Edition, 1999)

The concept of both views and the application of enterprise architecture frameworks can be demonstrated by the image featured on the cover of the 1979 book "Gödel, Escher, Bach" by Douglas Hofstadter. This image (Figure 2-5) features a complex three-dimensional object that is illuminated by three lights in different positions. From one perspective, the object's shadow forms the letter "G." From the other two perspectives, the object's shadow are the letters "E" and "B." The object itself is neither a "G", "E" nor "B." It is a unique, fairly "complex" three dimensional object whose true shape is the intersection of he shapes of he letters "G", "E" and "B" in three dimensions.

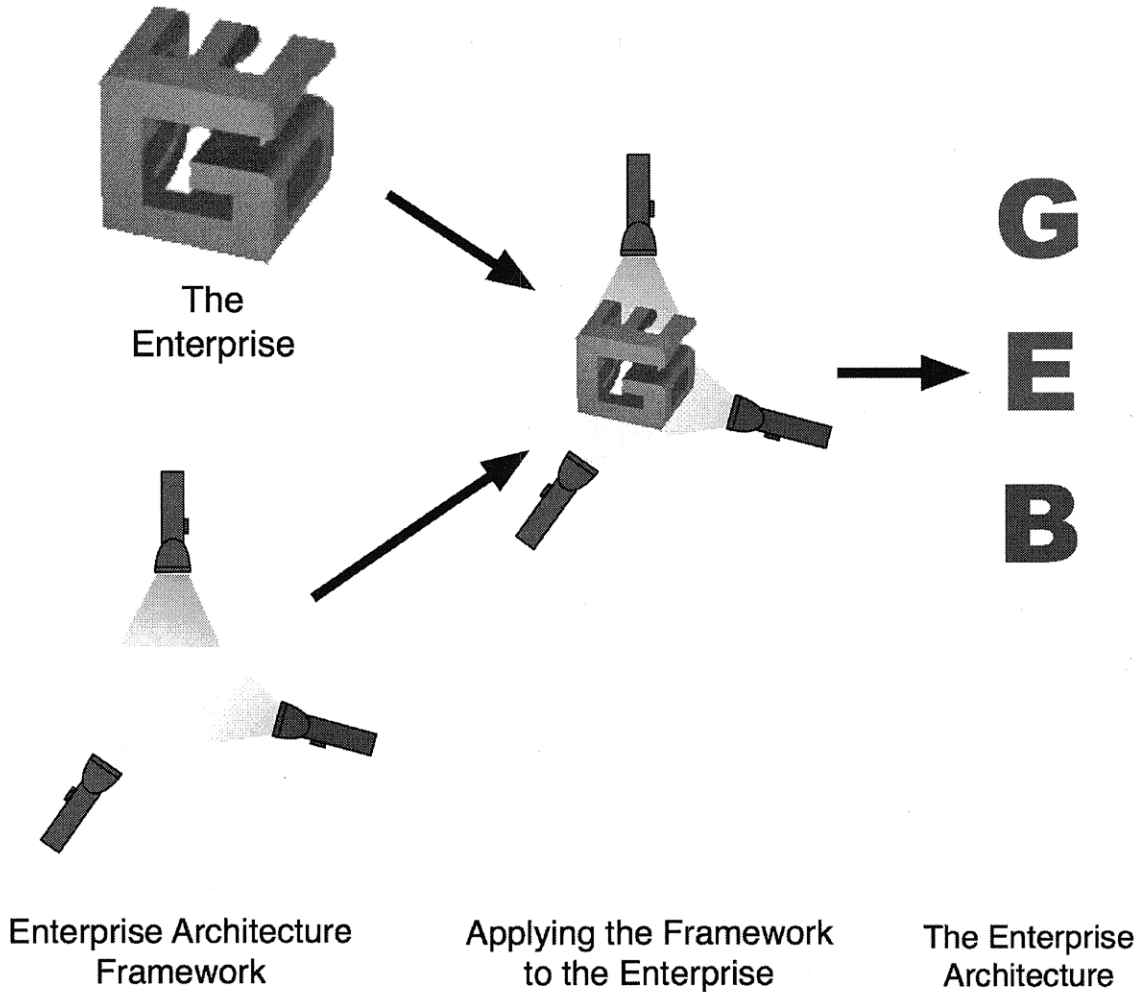


Figure 2-5: Using an architecture framework with multiple views to reduce apparent complexity

This image can be used as a simple analogy to understand the relationship of enterprise architecture, enterprise architecture frameworks, and the views of a framework. The complex woodblock can be used as an analogy for the “real-world” enterprise that the architect is attempting to characterize with an architecture. To help the viewer better understand the shape of the woodblock, a set of flashlights arranged in a specific manner around the block is employed. The number of lights used and their position around the object is important; if the lights are rearranged, a different set of shadows will be cast that may not contain

as much useful information. The set of flashlights and their arrangement can be thought of as analogous to an enterprise architecture framework; each flashlight is similar to one of the views employed by the framework. When turned on, each flashlight will cause the woodblock to cast a shadow. The shadow caused by each flashlight is an abstraction of the woodblock in a single dimension. Taken together, the collection of shadows constitutes an "architecture" of the woodblock. In this case, a woodworker could reproduce the shape of woodblock with only the knowledge of its "architecture" and the views used to create it.

There are weaknesses in this analogy, however. An enterprise is considerably more complex than a three dimensional woodblock. First, as previously noted, there is no way to know how many dimensions are needed to describe an enterprise. There is no way to prove if four or seventeen "flashlights" should be used to describe an enterprise. Second, there is no way of proving what any of the views should be. In the case of the woodblock, because it is a three dimensional convex hull, we know that we can use three orthogonal projections to completely describe it. Further, the lights were positioned in such a way that they were able to illuminate all of the concave features of the object (imagine if all of the flashlights were moved 30 degrees off from their positions in Figure 2-5; the resulting shadows would likely be blobs, rather than letters). In the case of an enterprise, there is no mathematically rigorous way to define the dimensions or even their relationship to each other. Unlike the flashlights, views used in enterprise architecting deeply interact with each other. They are not independent. Change in one enterprise view has the potential to affect other views.

The general failure to capture interactions among the various views in these reference architectures remains a serious intellectual and practical problem, limiting their usefulness for real-world enterprise transformation efforts. Perhaps unsurprisingly, many recent enterprise architecting efforts that have placed an emphasis on developing a detailed, micro-level documentation of the contents of the architecture within each view, instead of an emphasis on understanding of

the interaction of the views, have failed to meet the expectations of their stakeholders¹⁴. For enterprise architects, there is a balance of information that must be captured in an enterprise architecture. Too much detail within each view, depending on how the architecture will be used, may be wasteful; too little information may miss capturing critical components. Documentation of structure and interactions within the views must be balanced with documentation of the structure among the views. As with the architecting of buildings, enterprise architecting contains as much art as engineering.

2.3.3 The Use of Views in Popular Enterprise Architecture Frameworks

There is no commonly agreed upon number or set of views universally employed to describe enterprise architectures; enterprise architecture frameworks currently in use have been developed based on best practice rather than from a theoretical foundation. GERA identified 10 generic views grouped into four major categories:

- *Content Views*
 - Function
 - Information
 - Resource
 - Organization
- *Purpose Views*
 - Customer Service and Product
 - Management and Control
- *Implementation Views*
 - Human Implemented Tasks
 - Machine Implemented Tasks
- *Physical Manifestation Views*

¹⁴ While there are no academic journal articles citing failure rates of enterprise architecture efforts in industry and government, a search of Internet blogs reveals a somber picture. In one typical account given by Ivar Jacobson on BuilderAU, an enterprise architects forum, "Most EA initiatives failed. My guess is that more than 90% never really resulted in anything useful." He attributes the majority of failures to a heavy reliance on documentation over understanding and "gaps between the [views]." <http://www.builderau.com.au/blogs/syslog/viewblogpost.htm?p=339270872>

- Software
- Hardware

Despite this effort to identify a “fine mesh” list of possible views, few enterprise architectures in popular use employ the taxonomy of views above because new frameworks continue to identify different ways to decompose enterprises that provide value.

Table 2-2 provides a list of the views employed by some of the most widely used enterprise architecture frameworks. As can be seen, there is a very strong bias in current frameworks towards views that emphasize the role of information systems in the enterprise, rather than taking a broader perspective of the enterprise.

Table 2-2: A table of views employed by popular enterprise architecture frameworks.

Enterprise Architecture Framework	Views Employed
<p>CIMOSA Computer Integrated Manufacturing- Open System Architecture</p>	<ul style="list-style-type: none"> • Functional • Information • Resource • Organization
<p>Zachman Framework</p>	<ul style="list-style-type: none"> • Customer • Owner • Designer • Builder • Worker
<p>Federal Enterprise Architecture</p>	<ul style="list-style-type: none"> • Performance Reference Model • Business Reference Model • Data Reference Model • Services Reference Model • Technology Reference Model
<p>TOGAF (The Open Group Architecture Framework)</p>	<ul style="list-style-type: none"> • Business Architecture • Application Architecture • Data Architecture • Technical (infrastructure) Architecture

Each framework employs a unique collection of views to describe the enterprise in a way that suits the objectives of the framework. Some frameworks will “matrix” views; for example, the Zachman Framework has six “viewpoints”¹⁵ matrixed with six areas of focus to create a thirty-six cell framework of areas to described, as shown in Figure 2-6. In another example of a matrixed architecture

¹⁵ Enterprise architecture frameworks do not use a consistent terminology for the concepts of views.

framework, both GERA and the CIMOSA framework contains three dimensions of description: the views, the enterprise lifecycle (identification/concept through decommission), and the level of specification (general, partial, and specific). The concept level of specification is common to all frameworks, but specifically highlighted by CIMOSA. The generic specification is the structure given by the reference architecture; the partial specification is a reusable description that can still be adapted, and the specific is an architectural description specific to a particular enterprise. See Figure 2-7.

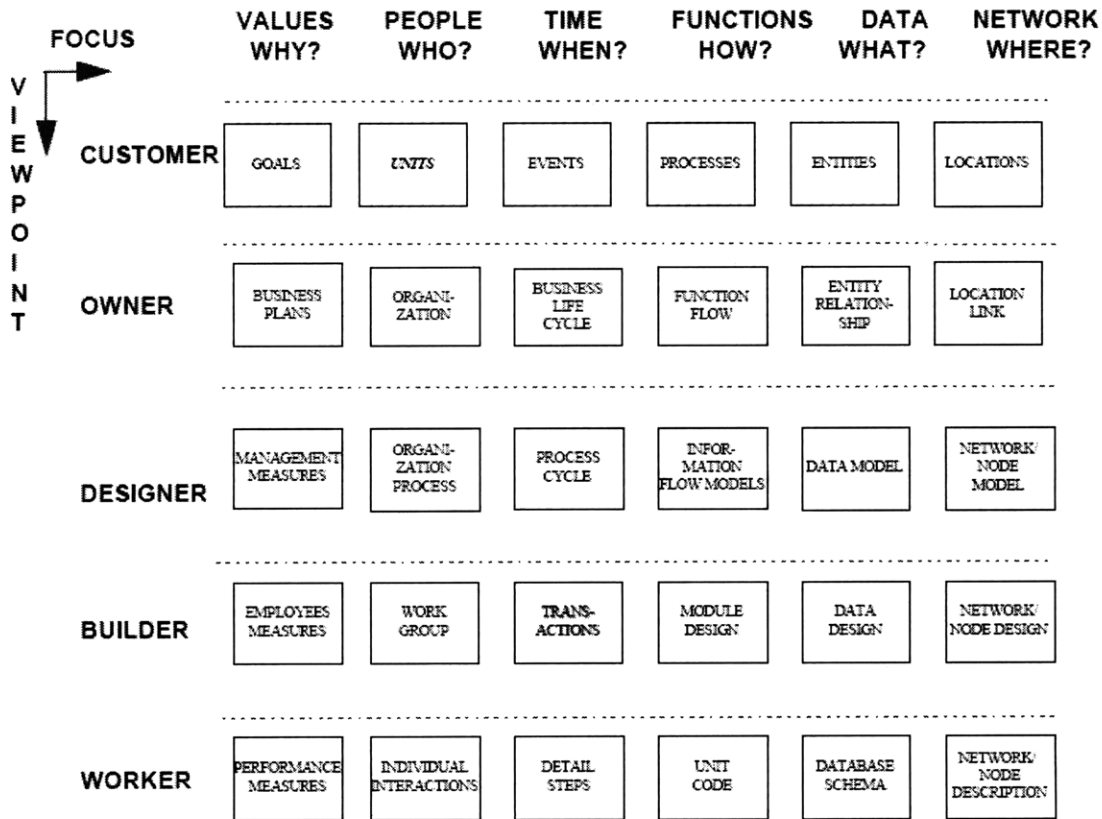


Figure 2-6: A simplified Zachman Framework Matrix¹⁶

¹⁶ Figure taken from <http://csrc.nist.gov/nissc/1996/papers/NISSC96/paper044/baltppr.pdf> As a work of the U.S. Federal Government, this image is in the public domain.

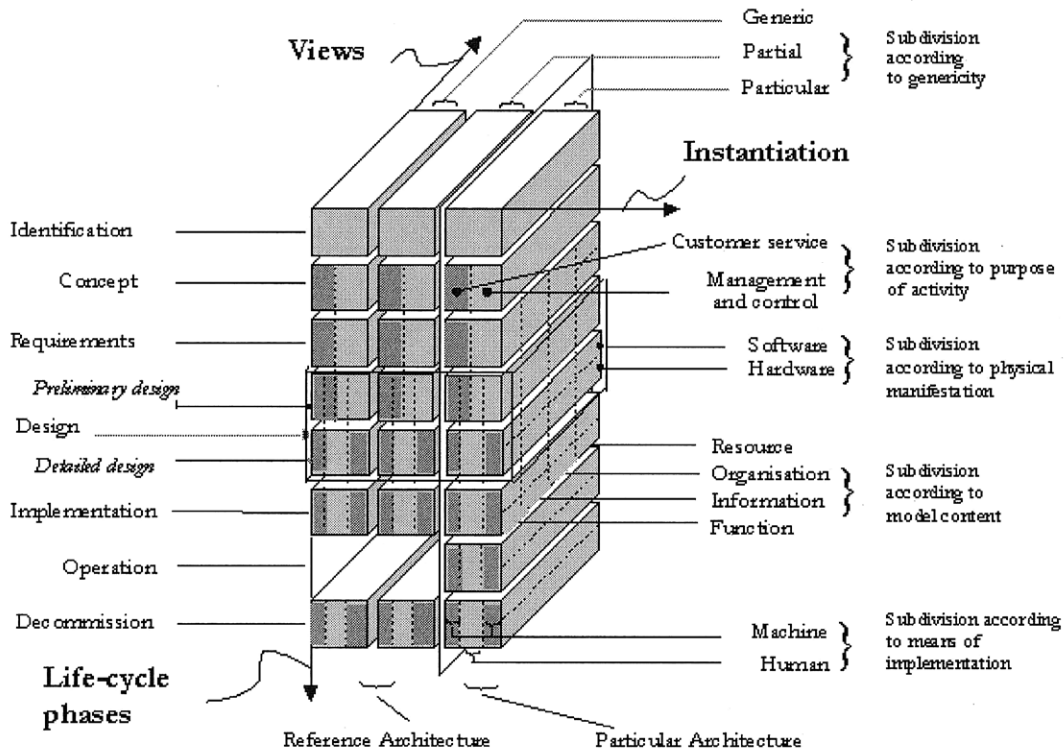


Figure 2-7: The GERAM/ ISO 15704 cube. GERAM, as a reference standard, identifies 10 possible views on the right of this figure. CIMOSA uses this representation with only four views: Functional, Resource, Organization, Information.¹⁷

A properly defined set of views should form a basis for complete description of the enterprise; that is, the views can be thought of as a “spanning set” of the enterprise, to borrow the term from linear algebra.¹⁸ As with the mathematical notion of spanning sets, there can be multiple sets of architectural views that can describe an enterprise, so long as they all are capable of describing all relevant aspects of the enterprise. Just as a physicist may switch coordinate systems

¹⁷ Figure from GERAM 1.6.3 / ISO 15704.

¹⁸ Linear algebra defines a set of vectors, $S = \{v_1, v_2, \dots, v_n\}$, to be a *spanning set* of a vector space V over a field K if by combination of the vectors in the form $\{\lambda_1 v_1 + \lambda_2 v_2 + \dots + \lambda_n v_n \mid \lambda_1, \lambda_2, \dots, \lambda_n \in K\}$ any point in V may be reached. A spanning set is not unique to a vector space. The vectors in S need not be linearly independent (if they are, they are referred to as a *basis* or a *minimal spanning set* for the vector space).

from Cartesian to Polar in order to more easily describe a system given a task at hand, an enterprise architect may choose to use a different framework depending on how the enterprise architecture will be used (e.g., IT system integration, enterprise change, organizational alignment, strategic management).

There is no one, “best” enterprise architecture framework that is best suited to all enterprises in all environments and for all applications. Each framework has its strengths and weaknesses, and is biased towards certain applications. As a result, it is not uncommon for enterprises to create “blended” frameworks that borrow best practices from a number of frameworks (Sessions, 2007). For example, Lockheed Martin has considered using a blended approach that uses the viewpoints of the Zachman Framework, the process components of TOGAF, and some of the reference architectures present in FEAF.¹⁹

For a comparison of popular enterprise architecture frameworks, including strengths, weaknesses, and suitability for different purposes, see (Sessions, 2007; Schekkerman, 2006).

2.3.4 Interactions Among Views

Most enterprise architecture frameworks currently in use either explicitly or implicitly assume that the enterprise architecture can be described by considering the independent contributions of the set of views that it espouses. Zachman, when establishing the use of views in his framework, noted that “each of the different descriptions has been prepared for a different reason, *each stands alone, and each is different from the others.*”[emphasis added](Zachman, 1987). While Zachman notes that the descriptions are “inextricably related,” he maintained that the documentation of each view must stand on its own.

¹⁹ This was presented to MIT’s ESD.38J Enterprise Architecting course by members of Lockheed Martin’s enterprise architecting team, April 2007.

Most enterprise architecture frameworks employ a partitioned graph of the various views that each framework uses to describe the structure of an enterprise, as shown in Figure 2-6 and Figure 2-7. The partitioned nature of this style of presentation implies that the views of a framework are independent of each other. In practical application of enterprise architecture frameworks over the past two decades, the views are often treated as *logical partitions* of the enterprise itself; that is, that there is no overlap in content between views. Each view is considered a self-contained description of an aspect of the enterprise, and when the full set of views are taken together in some way, current practice holds that these views constitute a “complete” description of the enterprise architecture. The net result is a plethora of enterprise architecture frameworks in use with essentially independent views.

There is a fundamental flaw with this approach to enterprise architecture: the views of an enterprise architecture are *not* independent. The views cannot be logically partitioned. In reality, each view is dependent on structures and behaviors in many other views, and overlap between views essentially always occurs. Behaviors and structures captured in any one view have cascading affects that cross the boundaries that exist between the definitions of views in *any* framework. For example, changes to an organizational structure *do* affect processes; changes to processes *do* affect knowledge requirements and information needs. While most users of architecture frameworks would agree that enterprises are complex, adaptive systems with a high degree of interdependency between their components, too many then ignore many of the conceptual interdependencies that exist when creating or documenting an enterprise architecture, despite the fact that enterprise architecture should be a tool to battle this very complexity.

As complex systems, enterprises are nearly-decomposable, not fully decomposable. According to Simon’s notion of near-decomposability, over the short-term, the behavior of the subsystems will exhibit approximate

independence. Over the long-term, however; the behavior of any subsystem is dependent in an aggregate way on the behavior of the other components(1962). The interactions between the major subsystems are significant drivers of enterprise behavior at these longer time scales and must be accounted for in any holistic analysis of system behavior.

2.3.5 The Nightingale-Rhodes Enterprise Architecture Framework

The Nightingale-Rhodes Enterprise Architecture Framework (NREAF), currently under development at the Massachusetts Institute of Technology, is a new framework intended to mitigate the traditional weakness of enterprise architecture frameworks from the perspective of management (Nightingale and Rhodes, 2009). First, the NREAF decomposes the enterprise using a set of views that intends to capture the enterprise holistically, rather than from the perspective of IT system integration or any other domain specific application. It specifically includes views drawn from the literature on organizations often ignored by other frameworks, such as knowledge, products, and external environments and policy. This decomposition chooses to focus the attention of the architect on high-level structures of the enterprise, rather than on a detailed analysis of operational attributes that is often found in other frameworks. See Table 2-3: The views employed by the Nightingale-Rhodes Enterprise Architecture Framework for a listing of the decomposition of views that it employs. For each view, the architect must describe both the associated structures and behaviors.

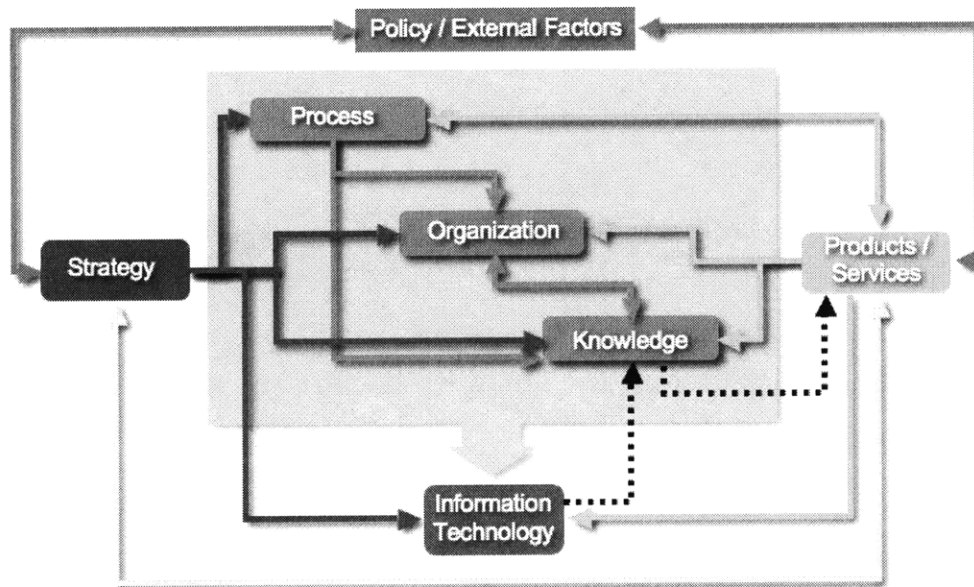
Table 2-3: The views employed by the Nightingale-Rhodes Enterprise Architecture Framework

View	Description
Strategy	The goals, vision and direction of the enterprise and includes the business model and competitive environment
External Factors and Policies	The external regulatory, political and societal environments in which the enterprise operates.
Process	The core, enabling and leadership processes by which the enterprise creates value for its stakeholders.
Organization	The organizational structure as well as the relationships, culture, behaviors, and boundaries between individuals, teams and organizations.
Knowledge	The implicit and tacit knowledge, capabilities, intellectual property resident in the enterprise
Information	The information needs of the enterprise including the flows of information as well as the systems and technologies needed to ensure information availability.
Product	The product architectures of the enterprise.
Services	The architecture of the services of the enterprise, including services as a primary objective or in support of products.

SOURCE: Nightingale and Rhodes, Lecture 3, ESD.38J Enterprise Architecting, February 2009

Second, NREAF places a specific emphasis on the interaction among these views and how the interaction of these views can give rise to unanticipated enterprise behaviors. The framework highlights a handful of key interactions among views that research has shown to be worth careful consideration and analysis when architecting. These interactions are shown in Figure Figure 2-8,

repeated from Figure 1-1. Primary interactions are shown as solid slides, while important secondary interactions (interactions in response to primary interactions) are shown with dotted lines.



SOURCE: Nightingale and Rhodes, Lecture 3, ESD.38J Enterprise Architecting, February 2009

Figure 2-8: The Views and Interactions in the Nightingale-Rhodes Enterprise Architecture Framework

In addition to explicitly identifying key inter-view interactions, Figure 2-8 can be used roughly as a map for the proposed architecting process, beginning with the an understanding of the external environment/ policy considerations and the development of a strategy and business case. Together with the products and services that the enterprise provides, these views inform the creation of architectures for the enterprise's processes, organizational structure and characteristics, and knowledge, in that order. These three views, taken together, should be used to inform the development of information technology.

Although the Nightingale-Rhodes Enterprise Architecture Framework is still under development, it offers to aid enterprise architects who seek to apply the practice of enterprise architecting in a more holistic and management oriented direction,

and provides a viable alternative to traditional frameworks.

2.3.6 Application of Enterprise Architecture Frameworks for Enterprise Architecting

In contrast to the traditional architecture that creates blueprints for the construction of new buildings, enterprise architecting is most often used to support the efforts of enterprises already in operation that seek to change in some way. Enterprise architecting efforts begin with the “current state” enterprise architecture: a blueprint of the enterprise in its current form. The current state architecture serves as the baseline for all enterprise change activities. In an ideal environment, an enterprise will always have its current state architecture updated and available.

In enterprise change efforts, some change has occurred that has made the current state architecture undesirable. Perhaps the environment has changed, rendering old processes inadequate; perhaps the enterprise’s strategy has changed in order to become more competitive in the market place. When a change is desired from the current state architecture, the first step is to analyze the current state to identify where changes should be made to achieve the desired behavior for the transformed behavior. Currently, there are few rigorous processes for behavioral or performance analysis of enterprise architectures. In practice, many “analyses” consist of establishing carefully crafted narratives that link architecture to performance based on observation; the better analyses will involve an application of metrics to support the narrative. Value stream mapping (Rother and Shook, 2003) and business process mapping (Sousa, et al., 2002), are typical of the type of analysis conducted on enterprise architectures. Such activities can potentially be very useful and are accessible to many organizations, but cannot provide a true analysis that links an enterprise’s performance to its architecture or measure how a change to the architecture will impact enterprise performance. As complexity in the enterprise architecture grows, the

effectiveness of such static, qualitative analysis techniques decreases.

Currently, the analysis of current state and future state enterprise architectures is a weakness of most enterprise architecting efforts. In some cases, analysis of the current state may be skipped entirely in favor of directly creating an alternative future state that addresses perceived problems. Such a weak treatment of analysis has the potential to completely undermine the value of enterprise architecting.

Despite the fact that enterprise architecture frameworks represent an attempt to provide a systems thinking approach to *describing* architecture, if the architectures are not also *analyzed* using a systems thinking approach they may prove to be ineffective. As a complex system, changes made to one part of the architecture may have unintended consequences elsewhere in the system. A proper analysis of the enterprise architecture will help us to understand where it can be decomposed and simplified, and where it cannot. Without a rigorous approach that can analyze the enterprise architecture and the behaviors it can potentially produce, the potential for unintended consequences is high and the utility of enterprise architecting as a management tool is almost negligible.

Some of the most promising approaches for analysis of enterprise architecture employ simulation techniques that model the behavior of the enterprise over time using the information captured by the enterprise architecture. The vast majority of attempts to simulate enterprise architectures have focused on simulating information systems, as enterprise architecture has disproportionately been focused on information system efforts and it is more straightforward to model the logical nature of information systems when compared to aspects of the enterprise architecture, such as organizational behavior or knowledge. Chapter Three will focus on simulation techniques that can be used to address the critical weakness of enterprise architecting.

After an analysis of the current state enterprise architecture has been completed, the information learned from it can be used to form hypotheses for how the enterprise architecture can be changed in order to produce more desirable performance. These hypotheses will be used to create one or more alternative “future state” enterprise architectures that have more desirable characteristics than the current state. Future state enterprise architectures may be more aligned with existing enterprise goals, aligned with a new set of goals, or may have other desirable qualities, such as flexibility or robustness to a changing business environment. Once the future state architecture(s) have been developed, they should be analyzed using similar methods used to analyze the current state architecture. If there are multiple candidate future state architectures, the analysis can be used to help make the selection²⁰. If analysis reveals potential weaknesses in the future state architecture, it can also be revised. Once a future state enterprise architecture is selected, the enterprise must develop a transition plan to transition from the current state to the future state architecture.

2.3.7 Utility of the enterprise architecture for understanding and managing enterprise behavior

Enterprise architectures are developed with the intent of describing an existing or future state of the enterprise for purposes of building a shared understanding and creating implementation plans. They are the product of a successful enterprise architecting effort. In addition to the traditional descriptive role, enterprise architectures have a potential to aid in the development of an approach to understand enterprise behavior. The advantages of enterprise architectures for understanding enterprise behavior are that they (1) provide frameworks which can be used to manage the structural complexities of the enterprise by providing a practical way of decomposing complex enterprises into discrete subsystems, or

²⁰ There may be many criteria used to make a final architectural selection, some at odds with each other. Performance can be measured on many dimensions; other desirable attributes include ease of implementation, flexibility, agility, and robustness, for example.

views, and that they (2) can serve as an integrative vehicle for other theories. Enterprise architecture frameworks are system-level frameworks for the study of enterprises as complex systems which make attempts to decompose the enterprise in ways that increase understanding of the enterprise's function and behavior while retaining key system-level interactions across the enterprise. When successfully implemented, enterprise architecture can provide a framework for the analysis of the enterprise from multiple perspectives, employing associated but disparate theories in a practical way. A common, agreed upon enterprise architecture framework is useful for creating a shared understanding of the enterprise.

Unfortunately, enterprise architecture as developed in practice does not often live up to the full potential of its claim or promise. Instead of serving in an integrative role, in practice enterprise architecture is often highly reductionist, despite its integrative intentions. The enterprise's architecture will be strictly decomposed, with the interconnections between views and subsystems often ignored, missing the value of understanding the enterprise as a nearly-decomposable system. Practical applications of enterprise architecture tend to have a focus on information systems with a high amount of detailed decomposition and a relatively low amount of insight. Enterprise architectures are typically created for information systems engineers and specialists, without as much input from high-level business stakeholders such as the CEO. As a result of this narrow, technical focus, most of the analytical tools developed for enterprise architecting are intended for analysis of information systems alignment, rather than other aspects of the enterprise, such as strategy, organization, processes, and knowledge.

Many of the analytical approaches often used in by the methodologies in common enterprise architecture frameworks have weak theoretical underpinnings, especially with regards to non-technical components of the architecture, such as the organizational design or strategy. Much work within the

enterprise architecting community has been invested in understanding the information needs of the enterprise as well as its processes their relationship to architecture, but there has been a disconnect between enterprise architecture and the vast array of literature in the field of organizational science in particular. To be of more use to enterprise leadership, enterprise architecture frameworks must be developed that broaden the scope of enterprise architecture away from information systems development and implementation.

An additional weakness of enterprise architecture frameworks is that they do not provide a theoretically rigorous approach to decomposing enterprise architectures. There are no tools for decomposing enterprises into a set of views based on a careful analysis of each individual enterprise's features. Instead, the reference architectures within frameworks provide a generic set of guidelines for decomposition. The decomposition employed by any one framework cannot be entrusted to be the one optimal possible decomposition for a given enterprise; rather, it is a "one size fits all" approach. In practice, reference architectures are tailored for use in a particular enterprise, adapted to its individual circumstances and structure.

Table 2-4 provides a review of the strengths and weaknesses of enterprise architecting as a framework for analysis of enterprise behavior and dynamics.

Table 2-4: A comparison of strengths and weakness of current approaches to enterprise architecting as a framework for analysis of enterprise dynamics

Enterprise Architecting	
<p>Strengths</p> <ul style="list-style-type: none"> • Manages structural complexity in enterprises through decomposition • Capable of integrating multiple enterprise views and associated domain-specific theories • Provides a system-level view of the enterprise by assembling multiple views • Widely adaptable; many enterprise architecture reference frameworks are available for use and adaptation 	<p>Weaknesses</p> <ul style="list-style-type: none"> • In practice, places undue focus on Information system integration rather than broader management concerns • Weak theoretical underpinnings guiding creation of many frameworks, especially organizational theories • Lack of tailored decompositions • Weak analysis methods available, especially for enterprise behavior

As a methodology, enterprise architecting is intended to be highly flexible and adaptable over a wide range of applications, providing both the ability to communicate an architecture as well as the ability to analyze how the architecture will behave. While enterprise architecting should be highly flexible in theory, in practice it has had a fairly narrow application IT applications within the enterprise. It has not served as a platform for more broadly understanding or describing the enterprise.

In order to use enterprising architecting as a tool for understanding broader enterprise behavior, an architect needs both a properly scoped framework with solid theory supporting analysis of the enterprise from multiple perspectives as well as analytical techniques that allow the analysis of the behavior and dynamics attributable to the enterprise architecture. To date, both the necessary framework and tools have been missing from the architect's toolbox.

The first step towards improving enterprise architecting's utility as a tool for understanding and designing dynamic architectural behaviors is to improve the theoretical underpinnings of the frameworks. In particular, most enterprise architecture frameworks, such as the Zachman Framework or the Federal Enterprise Architecture, provide a weak treatment of an enterprise's behavioral dynamics, with an eye towards understanding the interaction of an enterprise's organizational design with its processes, strategy, knowledge, and products and services. To improve upon this weakness, enterprise architecting must be supplemented. The organizational science literature can provide a rich collection of theories intended to aid the development and management of organizations. Together with existing research on other aspects of the enterprise such as processes, technology, and product and services, organization science research can provide the needed intellectual rigor necessary to create analysis techniques capable of understanding the behavior of enterprises, as it arises from the enterprise's architecture.

2.4 A SYNTHESIS OF ORGANIZATIONAL SCIENCE AND ENTERPRISE ARCHITECTURE FRAMEWORKS

A synthesis of organizational science literature with enterprise architecture frameworks can provide a more desirable approach to managing the complexity inherent in enterprise architecting over the application of either field on its own. From the perspective of the enterprise manager, both the theories of organizational science and the practice of enterprise architecture are useful, but neither constitutes a complete body of knowledge and tools necessary for an effective analysis of the complex dynamic capabilities of the enterprise. In many ways, however, these two disciplines complement each other. Organizational theories give rigor and add breadth where enterprise architecting is lacking; enterprise architecting provides integrative frameworks that can tie together

disparate theories for analysis from a systems perspective and can provide a structured approach and tools for creating the architecture.

Fortunately, enterprise architecture frameworks, if properly designed, can incorporate organizational design theories and present them in a coordinated fashion for application by the architect alongside other bodies of knowledge, such as the process, knowledge, and product architecture literatures. Enterprise architecture frameworks are intended to “facilitate the unification of methods of several disciplines... to allow their combined use, as opposed to segregated application” (GERAM, 1999). A synthesis of organizational theory with enterprise architecture frameworks will produce a workable analytical framework that can be used to examine the interactions of the architecture’s constituent components, and the relationship of the enterprise’s architecture to its behavior and performance. This synthesis can be achieved by incorporating organizational theory into the view(s) of the enterprise architecture that captures the organizational aspects of the enterprise’s architecture, expanding the scope of the views where necessary.

To date, none of the enterprise architecture frameworks in current use have strongly incorporated organizational theory into its set of views to any great extent, instead emphasizing information systems development and integration. Fortunately, shifting thinking within the field of enterprise architecting has led to the development of new, more broadly scoped frameworks that place a greater emphasis on organizational aspects of the enterprise and can potentially make greater use of existing organizational theory to include ideas from contingency theory and organizational design. The Nightingale-Rhodes Enterprise Architecture Framework, a new framework currently under development at MIT, shows potential promise as an architecting framework that can incorporate organizational theory to a greater extent, alongside traditional enterprise architecture foci such as information systems. Although still nascent, this framework has a much stronger organizational focus than any of the other

currently used enterprise architecture frameworks, while maintaining a depth in the description of processes, products, services, knowledge, and information systems that organizational design models such as Galbraith's or Nadler and Tushman's do not have. This framework will be described in detail in Chapter 4.

2.4.1 The need for Simulation Models

Enterprise Architecture frameworks, even with the inclusion of the perspectives and guidance of organizational theory, do not provide a complete toolset for architecting the enterprise. Enterprise architecture frameworks provide a series of lenses, theories and tools for reducing the apparent complexity of enterprises. A good framework decomposes the enterprise into manageable views that can be interpreted and understood both independently and in relation to other views. It does not, however, necessarily help managers understand how the enterprise architecture will behave in different environments with different inputs, or if the architecture will enable or preclude an expected level of performance.

An enterprise architecture created using a framework is a static description of the essential components of the enterprise and their interconnections. By itself, this static description does not provide enough information to analyze and understand the behaviors that a given enterprise architecture is capable of producing. Enterprise managers need an analytical capability, such a simulation modeling, to help analyze, understand, and compare enterprise architectures captured using a framework. As noted in Section 2.2.4, analytical capabilities are essential to the development of a true process for enterprise architecting (Levitt, 2004). Without such capabilities, there can be little assurance that any changes to the architecture will have their desired effect or that any understanding of enterprise behavior is correct.

Chapter 3 will address the need for architectural analysis tools by exploring the literature to identify appropriate simulation methodologies that can brought to

bear. Using the decomposition of enterprise frameworks and organizational science as guides, the contribution of different simulation perspectives will be explored, and a hybrid approach to simulation modeling will be developed that seeks to combine these perspectives in a way that is able to harness the strengths of these simulation methodologies while mitigating their weaknesses.

Chapter 3: APPROACHES TO SIMULATING ENTERPRISE ARCHITECTURE

Enterprises are complex systems with many interacting components in highly dynamic environments. Due to both their size and complexity, it is rare that a single person can understand all of their components, how they are structured, how they behave, and how their structure and environment affect that behavior. Enterprise architecting, supplemented with a solid understanding of associated theory, can be used to gain new perspectives on how the enterprise is composed and interconnected to form a coherent system. Unfortunately, this alone cannot help enterprise leaders better understand and anticipate how their structure drives their enterprise's behavior in a quantitative way. Enterprise leaders require a modeling capability that is representative of the real-life enterprise, captures the enterprise's behavioral complexity, and it is timely, adaptable and extendable. The "boxes and lines" models often used in enterprise architecting efforts, while valuable for communicative purposes, do not provide enterprise leaders with a quantitative behavioral analysis capability needed to develop a holistic perspective of enterprise behavior (Fowler, 2003).

Simulation modeling provides this capability by creating virtual worlds that can be easily manipulated and stepped through time to evaluate outcomes. They provide enterprise leaders with the ability to test their hypotheses of enterprise behavior and to evaluate alternative architectures and scenarios in an efficient manner. Some classes of simulation models, such as process simulations, supply chain simulations, and policy simulations have been around in some cases for decades. Unfortunately, these existing classes simulate the behavior of the enterprise within only a single view of the enterprise's architecture, missing the impacts of other views on enterprise behaviors. Enterprise leadership is currently missing the ability to simulate the architecture from a holistic, systems perspective capable of analyzing the effects of interactions not only within each view, but between the views. To gain this multifaceted understanding, hybrid simulation models employing a portfolio of simulation methodologies can be used to capture the behavior of the system from multiple, interacting perspectives.

This chapter will explore simulation modeling techniques that can be used to analyze enterprise behavior and will present the concept of hybridizing simulation techniques in order to capture the richness of different perspectives of enterprise architecture. It will begin with a discussion of current approaches used to model the enterprise, as currently used in enterprise architecting efforts. This is meant to motivate and highlight the need for simulation models that provide analysis not possible with the currently used descriptive models. After a discussion of why simulation models should be used, this chapter will explore what enterprise leaders need from simulation models if the models are to serve as a useful tool for management. These requirements will then be used to investigate three key simulation methodologies of interest: discrete event simulation, system dynamics, and agent based modeling. The chapter will conclude with an introduction to hybrid approaches to simulation modeling and a discussion of how a hybrid approach applied to enterprise architectures can meet the needs of enterprise architects, leaders and managers in their quest to understand how the enterprise architecture affects the behavior of the enterprise.

3.1 CURRENT MODELING METHODS USED TO SUPPORT ENTERPRISE ARCHITECTING

Models of enterprise operation are not new; they have existed for more than half a century. Simple logical and economic models of the operation of enterprises for the purpose of improving performance have existed since the advent of Taylorism in the early 20th century (Wagner-Tsukamoto, 2007). While there have been many approaches to modeling enterprises over the years, modern enterprise modeling as it is practiced in the enterprise architecting community has its roots in the functional modeling of Structured Design and Analysis (Ross, 1985), the informational modeling of Entity Relationship Diagrams (Chen, 1976), and data models of Design for Data Flow (Stevens, et. al, 1974).

The majority of enterprise architecture models currently in use are focused on *statically describing* the enterprise architecture in a manner consistent with a particular framework. These models are used to describe and communicate the structure the architecture, and are not intended to capture behavior. Commercial vendors will often offer “toolsets” that package together a set of modeling tools that can be used to create descriptive models that complement the decomposition and analysis favored by a given architecture framework. These tools provide a means of depicting the structure and relationships within an enterprise architecture, and do not adapt to individual views. The modeling methodologies used in the various enterprise architecture modeling toolsets have generally not been executable models, but rather graphical, “boxes and arrows” descriptions of the structure and relationships of the enterprise. Current modeling efforts have been disproportionately concentrated on capturing elements of information system architecture such as data interfaces and

structure, information flow, and alignment of information systems with existing processes (Bernus, 2003).

3.1.1 Examples of enterprise architecture modeling approaches in practice

Most approaches to enterprise architecture modeling currently used in practice employ a graphical descriptive methodology that identifies elements of the architecture and their relationships with one another. One such popular graphical modeling approach is the Entity Relationship Diagram (ERD), which define objects (entities), their relationships, and their attributes using a formal logic-based approach. ERDs are formal in the sense that the graphs can be translated into logical propositions that can be read by machines, or into natural pattern languages that can be read by humans (Bernus, 2003). Such diagrams and their translations are useful for implementing technical systems and aligning them to processes and business needs. They can identify disconnects in the logical assembly of systems, but seldom offer any insight into enterprise behavior or provide information that would be useful to enterprise leaders and decision makers. See Figure 3-1 for an example of a simple entity relationship diagram.

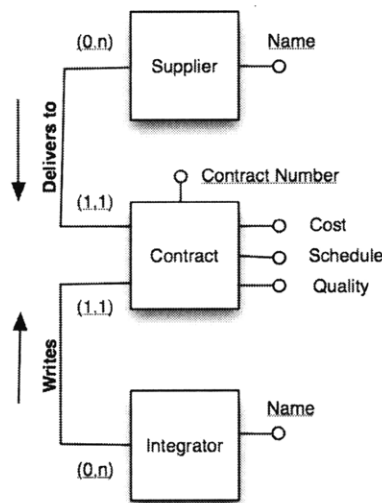


Figure 3-1: A simple entity relationship diagram that models a supplier-integrator relationship

The introduction of computing into the manufacturing environment was a driving force behind the introduction of the types of approaches outlined above in connection with the development and description of the various enterprise architecture reference frameworks. Computers, with their rigorously defined routines, required their related processes to be just as rigorously defined. As such, most enterprise architecture reference frameworks of the time were focused on understanding the enterprise architecture from the perspective of computer systems, rather than from the point of view of humans, entirely neglecting the underlying structure and behavioral dynamics of organizations and enterprises as complex sociotechnical systems. One such early language used to model enterprises was the Integrated Computer Aided Manufacturing Definition, known as IDEF²¹. As its name implies, IDEF is a series of specifications developed in the late 1970s by the U.S. Air Force Program for Integrated Computer Aided Manufacturing to help industry integrate computerization in to manufacturing processes (Colquhoun, et al, 1993). As such, there is a heavy reliance on defining process, information and data models. There are five IDEF specifications, each of which define a separate approach to modeling an aspect of enterprises:

- IDEF0: functional models
- IDEF1: information models
- IDEF1x: data models
- IDEF3: process models
- IDEF4: object oriented design models
- IDEF5: ontology description

The IDEF family of modeling languages can be considered the forerunners of current approaches to enterprise modeling, and continue to be used by some. IDEF has somewhat fallen out of favor for enterprise modeling as its methods are

²¹ IDEF is an acronym for ICAM **D**efinition.

not integrated and contain a significant amount of overlap in scope (Vernadat, 2001).

The heir to IDEF is the Unified Modeling Language (UML), adopted in 1997. UML is an object-oriented approach to system definition originally born from the software industry and its needs to document the structure and relationships between software components (Rumbaugh, et al, 1998). After proving successful in software development, UML was extended for use in business modeling and non-software systems. Like IDEF, UML offers a series of descriptions of a system from multiple perspectives that encompass how it is structured to how it will be used. It is a formal descriptive language, but does not possess any simulation capability. There are many commercial tools available on the market that make use of UML for enterprise modeling, such as Sparx's Enterprise Architect, IBM's Rational Rose and Microsoft's Visio. There are many other tools that exist that are built on proprietary schemes as well. Table 3-1 provides a list of some of the most commonly used enterprise architecting toolsets.

Table 3-1: Commonly used Enterprise Architecture Toolsets

Company	Tool	Website
Casewise	Corporate Modeler	http://www.casewise.com
IBM	Rational Software Architect	http://www-
IDS Scheer	ARIS Process Platform	http://www.ids-scheer.com
Metastorm	ProVision	http://www.metastorm.com/
Sparx	System Architect	http://www.sparxsystems.com.au/
Troux	Metis Product Family	http://www.troux.com/

Until the late 1990s, enterprise architecture “modeling” attempts, in the sense just described (i.e., as descriptive methods rather than as theory-based explanatory efforts), were exclusively focused on modeling enterprise activities, processes and functions using UML and IDEF, especially as they related to computer systems. The European Strategic Program for Research and Development in Information Technology (ESPRIT) Consortium, which authored the CIMOSA enterprise architecture framework, began to change this with its increased focus on business processes in its framework (AMICE, 1993). Traditional descriptive tools, which had been applied to processes, functions and data, were adapted for use on business processes, such as finance and program management. These tools largely remained static descriptions, however. One toolset that broke this trend was the ARIS (Architecture of Integrated Information Systems) Process Platform. ARIS, which focuses heavily on processes as applied to information systems, uses its own discrete-event simulation approach to capture processes (Scheer, 1992). Using ARIS, executable simulations of business processes could be created and analyzed with some support for strategic alignment, but the toolset retained an information systems perspective of the enterprise.

The Unified Enterprise Modeling Language (UEML) (Vernadat, 2002) was proposed to allow the sharing of information between enterprise architecture models, but its development has been slow and there has been little development activity on it in recent years. In that time, new products have been introduced into the marketplace, and the gap between them grows. Even if UEML were widely adopted, however, it would not necessarily improve the usefulness of these models to enterprise leaders. The same static, highly-detailed models could be linked together, but these models fundamentally are not capable of analyzing the dynamic behavior of the enterprise.

The adoption of enterprise modeling in support of enterprise architecture by business users has been limited. The tools of enterprise modeling are often not useful to them. The audience of early enterprise models was not the leadership

or other decision makers, but rather the computer systems and the people who had to implement those systems. This mismatch in audience persists, causing the management community to largely ignore the work done in enterprise architecting as “something for the IT department” rather than as a tool that can be useful to them as well.

3.1.2 The shortcomings of Current Approaches to Enterprise Architecture Modeling

All of these current approaches to creating enterprise models to support enterprise architecture suffer from a common weakness from the perspective of enterprise leadership and academia alike: the methods and models exist to support the implementation of systems (either information systems, new automation, new processes), not to support an understanding of how the design of the enterprise ultimately affects the performance of the enterprise. Regardless of the modeling approach used, current approaches are focused on capture of details of structure that serve as guidance for a person building or replacing a software system. The resulting models tend to be large, hard to maintain and complex to analyze due to the level of detail that they capture. As a result of the detail captured by these models, they can take quite some time to complete. It is not uncommon for the models to be out of date by the time that they are published, negating their value.

Additionally, these models tend to have a very narrow scope, limited to a single information system or process. As a result, there tend to be many “enterprise models” that are created as part of a single architecting effort. Maintaining the relationship between these models is extremely difficult, as they are kept separately and are not interconnected. Because they are typically created using different languages or tools, “interoperability is often near impossible, or requires intensive human intervention and ad-hoc decisions” (Bernus, 2003).

Table 3-1 provides a collection of proprietary tools matched to an enterprise architecture framework that should be able to interoperate. These tools are not interchangeable with other toolsets, however, and features are fixed by the vendor, limiting the user's flexibility to extend or customize tools.

An additional problem with the current state of enterprise modeling is that the many different models required by a single enterprise architecture framework cannot be integrated. Although many of the existing platforms are designed to complement the same enterprise architecture frameworks and may even use a common language, such as UML, these proprietary modeling packages are not interoperable. This has led to what Vernadat (2001) describes as a "Tower of Babel situation."

Current approaches to modeling enterprise architecture, while widespread, are not very useful to enterprise leaders who require a system-wide perspective of their architecture that is flexible and captures the behaviors in their enterprise that are driven by the architecture. Enterprise leaders would like to be able to ask "can this architecture increase my competitive position, and if so, how does it do so?" not "how will my payroll system interface into my other process infrastructure systems?" Fortunately, however, there are alternatives to these static descriptive models currently employed by enterprise architecture reference frameworks. Simulation models of the enterprise architecture that capture both structure and behavior of the enterprise, hold great promise.

3.2 WHY USE SIMULATION MODELS?

Simulation modeling has become a very popular analysis approach used by disciplines ranging from manufacturing floor planning to social sciences for several decades. A simulation model is a computer-executable representation of a system's behavior over time. Simulation modeling allows users to systematically investigate complex processes and behaviors in systems that do

not lend themselves well to traditional, closed-form mathematical analysis. Simulation models are relatively quick to develop, cost effective and flexible, aiding those developing generalizable theory as well as those seeking to understand the behavior of a specific enterprise (Carley, 2002; Fowler, 2003).

Most people tend to be poor at thinking about systems in a dynamic context, often extrapolating behaviors linearly. Unfortunately, behaviors in complex systems tend not to be linear; they exhibit feedback with delays and inertia (Sterman, 2000). Other behaviors may be driven by the interactions of individuals with localized incentives whose collective actions give way to an emergent behavior or property of the system. Such behaviors are often difficult or impossible to model using a closed-form mathematical approach, but lend themselves well to simulation.

Simulation models also can scale up easily, subject only to the constraints of available computational power. Similarly, they can be run very quickly, allowing timely feedback and a larger number of experiments to be run compared to data that could be collected through field studies. The rapid execution and ease of modification of simulation models also allows their use in “what if” scenarios that help users understand the effects of changes in the structure or environment of the system.

Simulation models allow a modeler to tackle complexity in a holistic fashion, in keeping with the tenets complex systems thinking outlined in Chapter 2. Simulation models are capable of simultaneously addressing multiple interactions between sub-systems across the enterprise. This allows the enterprise to be treated in its entirety, avoiding a “stove piped” treatment of its parts and their interactions. The ability to scale well as well as the ability to simultaneously measure and evaluate system-wide interactions have led complexity researchers to turn to simulation models of systems as their tool of choice (Simon, 1990).

Simulation models of social systems, such as enterprises, hold particular value. It is very difficult to conduct an ethical, controlled experiment on organizations when undesirable outcomes can impact the lives of those within the organizations. While social experiments can be conducted on small groups in a laboratory setting, the results often do not scale well up to the organizational level, and scaling such experiments into larger, more realistic groups is often too cost prohibitive (Carley, 2002). For this reason, social scientists have often sought out natural experiments—conditions found naturally that mimic a controlled experiment. These natural experiments are rare, however, and can only be analyzed in retrospect. No modifications can be made, and rarely do they isolate the exact mechanism a researcher would like to investigate. Simulation models allow a modeler perform experiments on an organization in an ethical fashion, and flexibly adapt the model to test multiple hypotheses. They can be used to test extremely rare or extreme situations, such as market collapse or terrorist attacks, or those that do not exist, such as the impact of proposed legislation (Carley, 2002). Simulation modeling allows for testing and evaluating complex systems without humans in the loop, greatly reducing the difficulty involved with experimentation.

While there are many reasons that simulation models can prove to be useful, the success of any individual modeling effort is dependent on establishing clear goals and objectives for the model. It is critical to engage the model stakeholders to determine what they hope to achieve by using a simulation model, and to ensure that the model created accordingly. While simulation models of processes and functions of the enterprise have been in use for decades, they have been aimed primarily at process owners and middle management—not at enterprise leadership at the highest level, who strategically guide the enterprise's architecture. In order to create simulation models that will be useful to enterprise leaders, modelers must take the time to examine what enterprise leaders might most value, and seek to orient the development of their models around these values.

3.3 WHAT DO ENTERPRISE LEADERS NEED FROM ENTERPRISE SIMULATION MODELS?

To begin the investigation of what simulation methodologies can be useful to enterprise leaders, it is first essential to examine what they need from the models. Enterprise leaders need set of trusted tools in their virtual toolbox that will allow them to understand the *behavioral* effects of their enterprise architecture operating in a complex, changing environment in a timely manner. Unfortunately, the graphical models of enterprise architecture addressed in Section 3.1 only *describe* the structure and relationships in the architecture—they do not capture or analyze behavior driven by the enterprise architecture. Graphical models are most useful to those implementing technical systems described by the architecture, largely not useful to the leaders and architects of the enterprise who need a tool to help them manage and guide the development of the enterprise through an uncertain, changing environment. Currently, enterprise leaders and architects largely lack such modeling tools that are sufficiently well-proven and can provide the desired level of resolution to help them understand enterprise behaviors that result from existing or proposed structures, incentives, and policies. They need the ability to create *simulation models* that can capture the dynamics of their enterprise over time.

Enterprise leaders have many requirements of simulation models that go well beyond the need of many highly abstract simulation modeling activities typically undertaken for academic purposes. Enterprise architecting is a real-world endeavor, focused on understanding and producing results relevant to a particular enterprise. As such, the simulation tools of the enterprise architect must be practically focused, easily understood by many stakeholders, timely, and relevant to the particular enterprise.

Before surveying the field of simulation methodologies that enterprise architects can bring to their aid, it is useful to first ask what they need from a simulation methodology. There is wide array of simulation methodologies that exist, from 3D workflow models to highly abstract “toy” simulations of hypothetical constructs, and not all methods lend themselves to supporting enterprise leaders. To support enterprise leaders, a simulation model must be:

1. Representative of the actual enterprise, its structure, dynamics, and environment;
2. Able to capture behavioral complexity as it arises from the architecture itself;
3. Able to address specific problems the enterprise faces in a communicable and timely manner; and
4. Capable of quick adaptation to facilitate hypothesis testing and scenario analysis.

These four qualities stress that an effective approach must not only be capable of simulating enterprise behavior, but also be capable of incorporating new ideas and applying them in a real world environment in such a way that they can influence decisions made in the enterprise.

3.3.1 Representative of the enterprise

Any simulation of an enterprise architecture must be representative of the real-world enterprise, its structure, behavior and environment. This stands in contrast to “toy” models that simulate overly-simplified archetypical structures and behaviors, such as the NK models used in organizational science research (Kaufmann, 1993; see Rivkin and Siggelkow, 2003 and Ethiraj and Levinthal, 2004 for applications). This is key for ensuring the validity of the model for use in a management application and for achieving stakeholder buy-in. Each enterprise is a unique, complex adaptive system that has been architected and evolved to

meet the demands of its environment and strategy. While enterprise architecture *frameworks* are generic representations, enterprise architectures, and by extension any simulation models of the architecture, are unique representations.

Simulation models must be representative of a specific enterprise's architecture in order for its users to build confidence in the model and validate it. A simulation will lack both internal validity and credibility with its users if it does not simulate the specifics of the architecture of that enterprise. Without this credibility, the model will not influence architecting or management processes.

3.3.2 Captures the mechanisms underlying behavioral complexity

A simulation model for enterprise leaders must be able to capture the mechanisms that drive complex behaviors in the enterprise. It is not enough to simply be able to reproduce a complex behavior; enterprise leaders would like a model that also captures the mechanism, offering them insights into policy changes they could take to change the behavior in a manner that they desire. In a complex system such as an enterprise, mechanisms can be both top down, in the case of designed processes and procedures, as well as bottom up, in the case of the effect of incentives and rules on the behavior of individuals within the enterprise. Further, many mechanisms underlying enterprise behaviors may span multiple views of an enterprise: the ultimate performance of a process depends not only on an understanding of the process itself, but also is influenced by organizational incentives, strategic alignment, and knowledge requirements (Mingers and Gill, 1997). A candidate simulation methodology that can capture these behaviors from a system perspective and be useful to management must be able to handle mechanisms that span across enterprise views. Any effective modeling approach to capturing complex enterprise behavior must be able to capture a wide array of causal mechanisms.

3.3.3 *Specific and Timely*

Creating a simulation of an enterprise architecture that is both realistic and captures dynamic behavior is a difficult job. However, such a simulation will only be effective if it can be developed in a timely fashion and can be used to address *specific* questions about the architecture. This is not because enterprise leaders are necessarily impatient (although some are), but because the environment and the enterprise are constantly changing. The rate of change in the marketplace is increasing steadily. A simulation model that takes nine months to develop will usually be out of date when it can be used. The more timely the simulation is, the more likely it will be used by enterprise leaders who are architecting and managing the enterprise.

This has two implications. The first is that simulation models must exhibit parsimony. They must include key structures, interactions and behaviors from an architectural perspective, while overlooking elements of the design that do not contribute to enterprise-level behaviors and capabilities. The second implication is that each simulation must have a specific, intended purpose: it must answer a particular question asked concerning the architecture. A model or simulation of a complex system without such a guiding question will quickly become untenable, as it must grow in size to answer any possible question. The question a model seeks to answer serves to scope the model. Because model scoping is often key to its successful use, the development of a guiding question is extremely important.

The simulation must also be clearly understood by a wide array of stakeholders. The results of highly abstract simulations are difficult to convey to a general audience. If the simulation cannot be easily understood by all stakeholders, it will be difficult to involve stakeholders in model testing, and their confidence in the model's results will be lower.

3.3.4 Capable of Adaptation

The final attribute that enterprise leaders need from a simulation of enterprise behavior is that the model be adaptable. The environment in which the enterprise is embedded is quickly changing; any analysis of enterprise behavior must factor the uncertain environment into account. Model users must be able to quickly make changes to the model, allowing them to perform scenario analysis where different assumptions and structures can be quickly evaluated against each other. Enterprise leaders, in their struggle to learn about the balance between exploration and exploitation of their business environment (March, 1991), should be able to use such a simulation capability to understand the effects of different architectural choices with respect to these different enterprise goals.

These four attributes—representative, captures underlying mechanisms, timely, and adaptable—should be kept in mind while exploring the potential for simulation models of enterprise behaviors. While there are many potential simulation methodologies available for consideration, no single existing approach can address all four of these requirements. The strengths and weaknesses of each approach must be assessed to determine how it may be brought to bear in aiding enterprise leaders better understand how their enterprise architecture influences the behavior of their enterprise.

3.4 SIMULATION METHODOLOGIES FOR ENTERPRISE BEHAVIOR

Within the enterprise simulation modeling literature, there are three simulation methodologies that have been used to simulate enterprise behavior: discrete event simulation, system dynamics, and agent based modeling²². Each

²² While there are other simulation methods that have been applied to modeling enterprises in a theoretical setting in recent years, (e.g., NK models (Kaufmann 1993) or cellular automata (Wolfram 2002)), these approaches generally produce “toy” models, which are highly abstract. Because these approaches do not meet the requirement that behavioral models of enterprises be representative of sufficiently “realistic” (i.e., as opposed to highly abstract or highly simplified) real-world behaviors, these methodologies are not included in this discussion

methodology is descended from a specific field of study that has its own perspective of enterprise behavior. Discrete event simulations, for example, were born from the study of enterprises in an operational setting, concerned with coordinating and streamlining the flow of objects, as found in an assembly line or other repeatable process and has been used in existing enterprise architecting efforts. System dynamics and agent based modeling were born out of the study of organizations from a more theoretical perspective, with the older system dynamics taking a top down perspective of the enterprise's behavior, and agent based modeling capturing enterprise behavior from the bottom up. The next three sections will investigate the strengths and weaknesses of these simulation methodologies, and how they might be used to simulate behaviors driven by an enterprise's architecture.

3.4.1 Discrete Event Simulation for the Enterprise

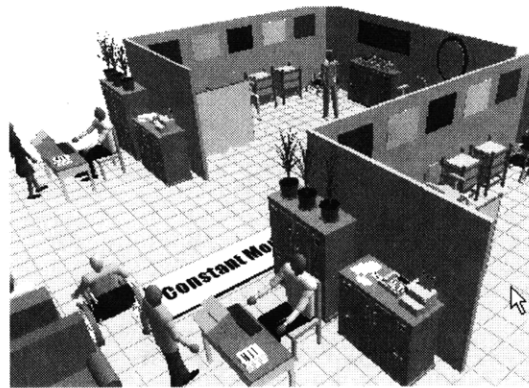
Discrete event simulation represents a class of simulation models focusing on the operation of a system as a series of chronologically ordered activities or events that can trigger a change of its state. These events occur at discrete points in time, determined either by predefined schedules or by probabilistically set timers. Discrete event simulation uses a "transaction-flow world view," where the system is viewed as constant of elements of traffic (originally termed as "transactions," although now more commonly referred to as "entities") that flow between points in a system, competing for resources (Schreiber and Brunner 2007). The path of the entity flow, as well as decision nodes and branch points, are depicted using a flow chart like block notation.

Entities are used to represent objects in a system, such as a vehicle or a person. Entities may take on properties, such as position, age, or color that can be modified by an event in the system, but typically do not exhibit artificial intelligence. Entities may move through the system, where they can be delayed, processed, placed in queues, consume resources, and change course as the

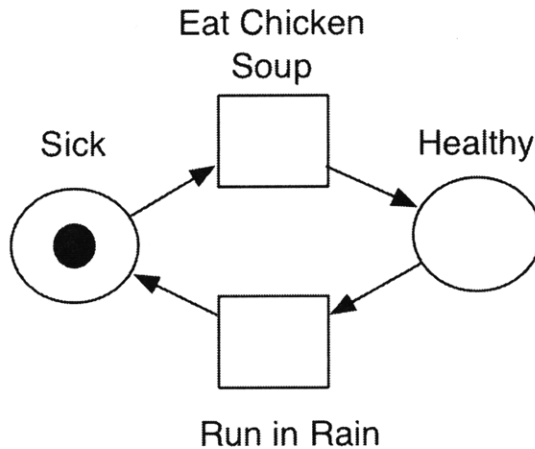
result of probabilistic events as they flow through a defined process path. An introduction to discrete event simulation can be found in Schreibner and Brunner (2007) as well as Chapter 10 of Cassandras and LaFortune (2007).

Discrete event simulation began as a simulation technique in the 1960s after the development of the General Purpose Simulation System (GPSS) at IBM by Geoffrey Gordon (Gordon 1961). Since that time, most discrete event simulations have followed the GPSS approach, using entities (called "transactions" in GPSS) that flow through a block diagram representation of a system set to a clock. Such simulations could easily gather statistics on processes, such as the average wait time in a queue for an entity, or identify bottlenecks in processes.

In recent years, a new class of discrete event simulation, called Petri Nets, has come into common use. Petri Nets simulate the transition of systems between places and transitions. Petri-Nets can be used both as a graphical notation and as a simulation methodology for systems. When described graphically, they can resemble flow charts with which many enterprise leaders are familiar. Petri Nets have become popular in the literature due to fact that their mathematical properties can be formally analyzed for completeness, freedom from deadlock, and reachability of all states using Markov Chain representations of the simulation (Peterson 1981). A review of Petri-Nets, along with extensions to the original formulations such as colored Petri Nets, can be found in (Murata 1989).



(A)



(B)

Figure 3-1: Examples of the extremes of discrete event simulations. A: A 3D animated simulation of a hospital process (www.systemflow.com) B: A simple Petri Net of a person's health.

Discrete event simulations can vary widely in how representative they are of real-world systems. At one extreme, three-dimensional animated simulations of processes created in software packages such as Arena[®] create life-like representations of system in time, logic, and space. Figure 3-1 A shows an animation from a discrete event simulation of the patient intake process at a doctor's office. Here, patients move through a hospital setting based on predefined transitions from one state to another, such as "waiting," "seeing

nurse,” and “seeing doctor.” In this simulation, these states consume resources (e.g., seats, doctors, nurses) and are associated with a spatial position in the simulation. The patients form queues as they wait for resources to become available. Such highly realistic simulations are useful for detailed implementation of complex, spatial processes. Most discrete event simulations do not need such high resolution, however.

At the other extreme of system description are Petri Nets. Petri Nets use a “token” (a highly simplified entity) that moves between states, shown as circles, through transition events, shown as squares. As opposed to the “transaction flow” view of the system taken by most discrete event simulations, Petri Nets take was is termed a “place/transition” view: emphasis is placed on the state and transition path, rather than on the flow of a number of entities. Figure 3-1 B shows a very simple Petri Net of a hypothetical person’s health. The black dot is the token, which in the case, represents a person. In the figure, the token is in the place “sick,” indicating that the person is in that state. The person can transition to “healthy” using the transition “eat chicken soup.” This transition may be triggered using a timer or may come from an external event (e.g., user intervention, phone call from mother). The person will remain in the “healthy” state until they transition by “running in the rain.” Each transition can be deterministic or probabilistic. These graphs and their transitions can be concurrent, asynchronous, distributed, parallel, non-deterministic, and/or stochastic (Murato 1989). Despite their high level of abstraction and forced formalism, Petri Nets can be used to create fairly representative models of processes that have been used to model enterprise processes.

Uses in the enterprise

Discrete event simulations have long found application within the enterprise, primarily to model various enterprise processes, ranging from Just-In-Time factory floor production (Huang, et. al., 1983) to higher-level business processes

such as new product development (Kalpic and Bernus 2002) or engineering change orders (Sousa, et al. 2002). In the late 1990s, Petri Nets were used to help model information business process (Nüttgens et al. 1998), and were incorporated as a tool within the ARIS enterprise architecture toolset. As a result, Petri Nets have seen limited adoption in industry for process modeling (Scheer, 2000). These models are typically intended for use by process managers seeking to redesign, analyze, or optimize processes, and have not been intended for use as management tools.

Strengths

Discrete event simulations have been used to model enterprise processes for several decades because they have proven themselves to be highly useful, capable of helping process owners better understand their processes in order to improve them. Discrete event simulations are often easy to understand from both a theoretical and practical perspective due to their relative simplicity and the ease with which they can be represented using a process notation such as UML. They have a large user and developer base within industry. Because they are often represented graphically as flow charts, anyone familiar with the system can quickly understand how the model operates and can interact with it. As such, they can serve as an effective communication tool, enabling a shared understanding of system operation.

Discrete event simulations are particularly useful for analysis of process performance, such as identification of process bottlenecks or collection of statistics on the process performance. They can be used to determine the average, minimum, and maximum time an entity can take to travel through a process or wait in a queue for further processing, for example. Variables within the simulation, such as average servicing time for a service, or the expected pass rate of an inspection process, can be borrowed from metrics taken in the real-world system, or varied to determine the effect on system performance of improving service times or pass rates. This makes discrete event simulations

very representative and easy to calibrate for processes that can be easily measured.

Weaknesses

While discrete event simulations are powerful, they do have limitations. The primary drawback is that they can only model systems using a “transaction flow” or a “place/transition” worldview. Many functions of the enterprise, such as strategic management or knowledge management, do not exhibit transaction flow behavior that feature some form of entity flowing along a path or transitioning between states, and could not be modeled using discrete event simulation. Discrete event simulations are often very computationally intensive for larger processes, and in some cases, can be modeled more efficiently using a continuous, equation-based approach (Borshev et al. 2002). The processes and the variables within a discrete event simulation must be completely specified before the model is executed—there is no adaptation of the model structure during runtime. Similarly, the entities in discrete event simulations have no autonomy—they do not make decisions, adapt, or learn, but simply follow a process and are acted upon. For this reason, discrete event simulations are typically not used to model social systems or decision-making, but are instead used to model stable, repeatable processes.

Discrete event simulation has a long history of application within the enterprise, with a particular focus on process modeling. Large libraries of common processes and functions have been built to speed the development of process model, which make creating discrete event simulations of processes particularly easy. This is the application for which it is uniquely suited, and it is unlikely that any other simulation methodologies will displace it in this role in the foreseeable future due to its widespread application in industry and ease of use in application. That said, discrete event simulation is not optimal for many aspects of an

enterprises' architecture that does not within the entity flow or place/transition worldview, such as strategy development and organizational design.

3.4.2 System Dynamics Modeling

System dynamics is a continuous, equation-based simulation approach developed by Jay Forrester in the late 1950s to model industrial behavior at a macro-scale. It is the “study of information-feedback characteristics of industrial activity to show how organizational structure, amplification (in policies) and time delays (in decisions and actions) interact to influence the success of the enterprise” (Forrester 1958). Borrowing principles from cybernetics and control theory developed in the same era, system dynamics is a simulation methodology that marries the power of continuous time differential equations with the clarity of diagrams that can be used to show the causal structure within a system and its effect on system behavior. From its inception, system dynamics was intended as a practical tool to effect change at the highest level of organizations:

“System Dynamics is an approach that should help in important top-management problems... The attitude must be enterprise design. The expectation is for major improvement. The attitude that the goal is to explain behavior, which is fairly common in academic circles, is not sufficient. The goal should be to find management policies and organizational structures that lead to greater success” (Forrester, 1961:449).

In the 50 years since its introduction, system dynamics has been applied to a wide variety of problems, particularly those that exhibit significant feedback and delays leading to complex and difficult to predict behavior, such as supply chains and resource allocation.

System dynamics uses simple diagrams to represent the causal dependencies in a system. These “causal loop diagrams” can be translated as a set of differential equations that can be solved for a solution given initial system conditions. In system dynamics, processes are presented in terms of “stocks” (e.g., people, material, knowledge, money), “flows” between the stocks, causal variables that influence the flows, and delays between the causal variables. Stocks, flows, and delays give the system inertia and memory, and can allow for the modeling of disequilibrium dynamics (Sterman 1997).

In system dynamics notation, stocks are shown as boxes, while flows are shown as pipes with valves on them, borrowing from a hydraulic metaphor. Causal variables are shown connected to flows and each other using arrows to indicate the direction of causality, with either a “+,” to indicate a positive relationship, or a “-,” to indicate a negative relationship, next to the head of the arrow. Figure 3-2 shows a simple model of an inventory system.

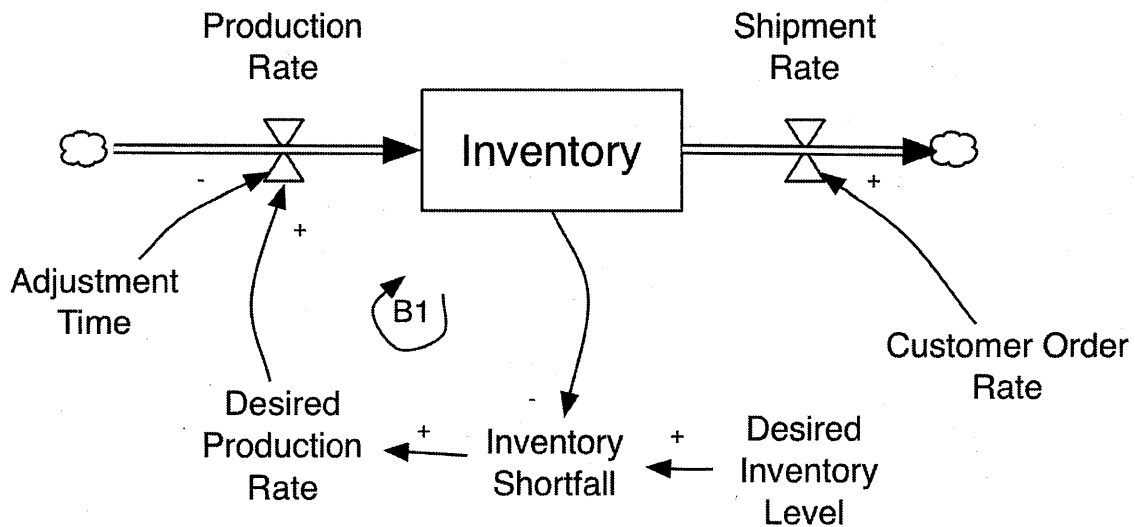


Figure 3-2: A System Dynamic model of an inventory system with a single feedback loop

In Figure 3-2, “Inventory” is a stock that is increased by the flow “Production Rate” and decreased by the flow “Shipment Rate.” It has a single “balancing” feedback loop, indicated by B1, which acts to adjust the production rate based upon the Desired Inventory Level²³. The differential equation for this relationship would be written as

$$\frac{dInventory}{dt} = Production\ Rate - Shipment\ Rate$$

where

$$\begin{aligned} productionRate &= Desired\ Production\ Rate / Adjustment\ Time \\ &= (Desired\ Inventory - Inventory\ Shortfall) / Adjustment\ Time \end{aligned}$$

Solving for the Inventory as a function of time, the system can be described as

$$Inventory(t) = Desired\ Inventory - (Desired\ Inventory - Inventory(0))e^{-t/AdjustmentTime}$$

where $Inventory(0)$ is the initial state of the Inventory. This is a very simple, linear first order negative feedback system that demonstrates an exponential decay. System dynamics software packages, such as VenSim, automatically generate the system’s equations based upon the diagram created by the modeler. First, second, and higher-order feedback systems can be modeled by adding in more feedback loops, which may be balancing (goal-seeking) or reinforcing (amplifying). Classic dynamic patterns of behavior, such as exponential growth and decay, S shaped growth, growth and collapse, and oscillatory behavior can all be generated using different combinations of feedback and delays.

²³ This is a very simple model, where the desired inventory level is treated as a constant. In realistic systems, the desired inventory level may be a variable dependant on external environmental factors and risk tolerance.

Unlike discrete event simulation, which processes entities and events at discrete time steps, the flow through a system dynamics model is continuous and aggregated into homogeneous stocks. For example, if a system dynamics stock measured the population of a city, the population would be measured as an aggregate value. The value of a stock may indicate a fractional value, such as 123.345, even when an entity in the model such as a person cannot be fractional. A particular entity cannot be traced through a system dynamics model, as it could be traced through a discrete event simulation. These differences provide both versatility and drawbacks when modeling systems compared to other simulation methodologies.

Uses in the enterprise

System dynamics has been applied to modeling enterprise behavior since its first application to modeling production and distribution dynamics in industry (Forrester 1958). It has been used both as a tool to create generalized theory (e.g., Reppenning 1997), as well as a tool for modeling specific systems for the purpose of forecasting and developing policy and design recommendations (Lyneis 2000). It has been used to study the dynamics of strategy (Fowler 2003), supply chains (Angerholfer and Angelides 2000; Scheiritz and Größler 2003), product development (Ford and Sterman 1998) and business processes (An and Jeng 2005), among many other areas. Sterman (2000) and Fowler (2003) argue for its use as a hands-on organizational learning tool (Senge 1990) to help managers discover and understand the structures underlying the dynamics of their organizations.

Strengths

System dynamics is a very powerful simulation methodology that is particularly suited to modeling the enterprise for high-level decision makers, as it captures the dynamics of the system from the macro-level, where they make their decisions. It is effective at capturing the “big picture” in a system, and methods have been developed using “causal loop diagrams” to explain model creation that help involve decision makers in the model building process, aiding both model

testing and helping to build confidence in the models (Sterman 2000). Often, the basic dynamics of a system under study can be roughly captured with only a few feedback loops; high fidelity models consisting of hundreds of highly calibrated loops typically are not necessary to build insight into a problem's dynamics.

System dynamics simulations have been used successfully as decision support models. System dynamics simulations typically can be run very quickly, enabling decision makers and model users to adjust model parameters and perform "what if" analyses in near real-time. The causal structure of the models also allows decision makers to quickly identify key variables and policy options that can act as levers into controlling system behavior.

These capabilities have led researchers to develop "management flight simulators" that present users with a dashboard-like interface with key metrics and control knobs, allowing them to "fly" an enterprise for a given duration and observe the results of management decisions on system behavior. Such models are often used to help teach systems thinking by building up simulated experience with complex systems (Sterman 2000; Fowler 2003).

Weaknesses

Although system dynamics is a very powerful tool, it does have its weaknesses. The primary weakness for modeling enterprise structures and behaviors is that it only works in aggregate terms—all people or resources are treated as a single homogenous resource that is varied continuously. This has computational advantages, but it poses challenges in modeling heterogeneous systems. The aggregation of resources/entities in system dynamics makes it easy to implement models of the high level structure of a system, but makes low-level modeling of systems challenging, compared with heterogeneous simulation approaches. System dynamics modelers have responded to this weakness by increasing the number of stocks used in their models, but this increases the amount of work for

the modeler and makes the model more complicated, harder to debug, and harder to explain to users seeking to understand the structure of the model. The reliance of system dynamics on aggregated resources also requires that the data used to calibrate these models must also be aggregated, which may prove difficult to collect in practice.

Because system dynamics fundamentally relies on modeling the causal dependencies in a system, the modeler must have a firm, mathematical understanding of each of these relationships with supporting data and metrics to model them. For many system dynamics models of “soft” processes, such as strategy or knowledge, it can be difficult to quantitatively capture these relationships, as the relationships may be uncertain. Uncertainty in the initial conditions of the model can potentially cause drastically different behavior, especially if the model is operating in a region of instability or disequilibrium. For this reason, sensitivity analysis of system dynamics models very important.

Additionally, it is important that the system dynamics modeler not overlook any key feedback loops and variables. The importance of many such loops may not be apparent upon examination of the system being modeled. Models of management systems, for example, must model concepts such as “schedule pressure,” which can lead to lower quality and then an increase in re-work, slowing production. Because of reliance on causal loops, missing such a loop in a system dynamics model can greatly alter its behavior. While this criticism could be said of almost any simulation methodology, it becomes a greater challenge when modeling “softer” systems that are often modeled using system dynamics.

System dynamics is best applied to modeling macro-level dependencies and dynamics in the enterprise. When properly applied, it can be a tremendously useful simulation approach, allowing the development of models that can build insight into a system’s behaviors and uncovering the key policy levers into a

system. For analysis of the micro- to macro- relationship, however, another simulation approach is needed for modeling enterprise behavior.

3.4.3 Agent-based Modeling

Agent based modeling, sometimes referred to as multi-agent systems (Weiss 1999) or artificial societies (Sawyer 2003), is a simulation methodology that employs populations of decentralized, autonomous software “agents” that operate in parallel and communicate with each other using internal rulesets to produce system-level behavioral patterns. Unlike system dynamics, agent based models capture the micro- to macro-connection in systems, simulating how the interaction of populations of locally directed entities with micromotives can give rise to global, macro-behaviors (Schelling, 1978). The behavior of an agent based model cannot be predicted or derived from the properties of the agents themselves; the only way to uncover the system behavior is by running the simulation (Gilbert, 1995).

Axelrod and Tesfatsion (2006) claim that agent based models are defined by two key criteria: (1) the system is composed of interacting agents, and (2) the system exhibits emergent properties, i.e., properties arising from the interaction of the agents that cannot be deduced simply by aggregating the properties of the agents. Agents are directed by their internal schema, which are rulesets that define their decision-making capabilities. Agents can have memory, may adapt and learn, can be spatially aware, and can take actions within the simulation such as changing their state (e.g., “hungry,” “green,” location) or making a decision to take an action based on input from their local environment. Agent based models are the only simulation methodology capable of modeling behavior of locally rational, micro- to macro- behavior.

The earliest models that today may be considered agent based models were cellular automata such as Conway's Game of Life (1970). Cellular automata are a simple class of agent based models consisting of cells in a two dimensional checkerboard arrangement that may change state (e.g., existence, color) at discrete steps in time using local decision rules. System behavior is entirely determined by the initial state of the system. Conway's Game of Life is a binary cellular automaton where cells "live" or "die" based upon a simple set of rules: a live cell with less than two, or more than three, live neighbors dies. A dead cell with exactly three neighbors becomes alive. Other cells do not change. While the rules are very simple, the patterns produced by the Game of Life can be quite complex: as the game is stepped through time, some patterns are stable, while others may repeat in place with a given period, and others may "travel" across the board. Figure 3-3 shows three iterations of the Game of Life beginning with a simple "X" pattern on an 11 x 11 grid. Black cells are alive, while white cells are dead. After three iterations, the "X" has transformed into an "O."

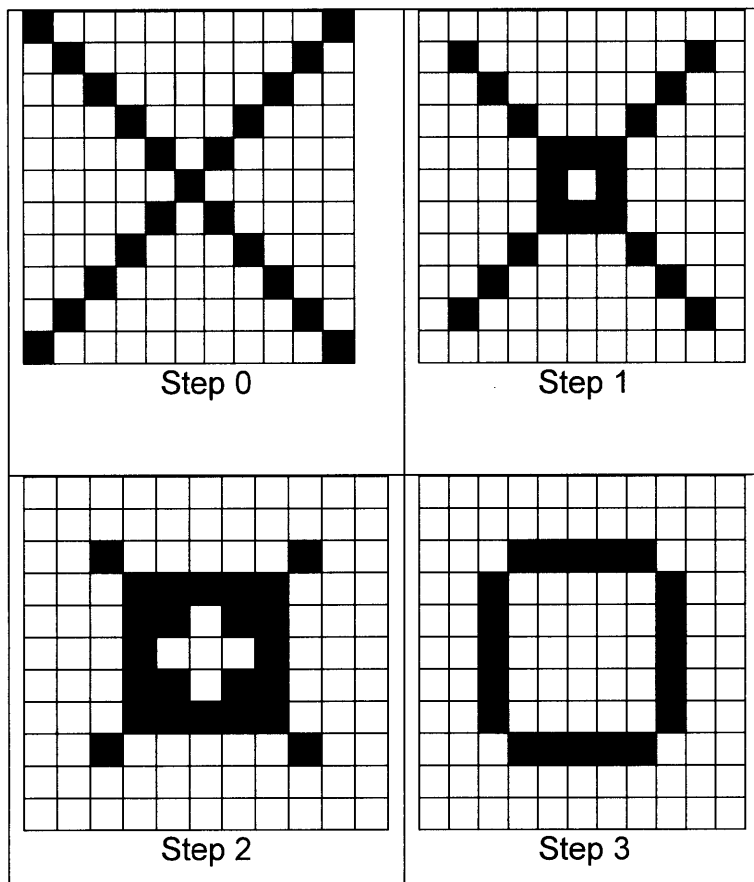


Figure 3-3: The Game of Life, three iterations of a basic "X"

The BOIDS model, developed by Reynolds (1987), was among the first simulations to resemble modern, more complex agent based models employing agents with more advanced schema not bound in a strict cellular arrangement. BOIDS was used to demonstrate how complex flocking behavior in birds can arise based upon three simple rules that birds follow: separation, alignment, cohesion. In this model, users can actually see software "boids" flock together through a virtual world and observe the behavioral change as the boids' internal rules (e.g., cohesion constant, separation constant) are modified.

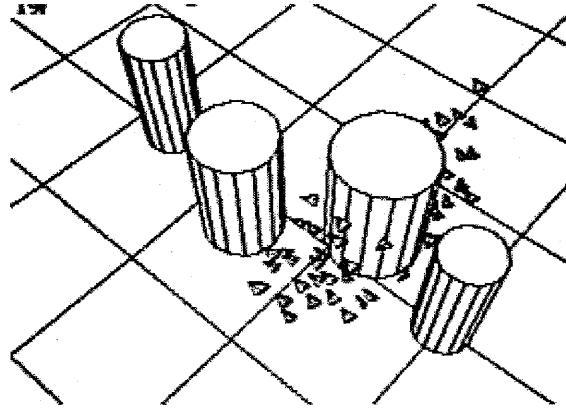


Figure 3-4: The Boids model, showing flocking behavior of birds around obstacles. From (Reynolds 1987).

After the development of object oriented programming languages such as C++ and Java in the mid-1980s and early 1990s, new software libraries were written to make development of agent based models easier²⁴. Agent based models began to be explored for use within the distributed artificial intelligence community (Sawyer, 2003). Much more advanced schema were developed, such as probabilistic and heuristic interaction rules. Agents could have many possible states, with highly developed transition rules between states.

Agent based models can study both how populations of agents interact individually, as well as how they collectively interact and respond to their environment. Agents may be used to explore unknown environments, using heuristic algorithms such as genetic algorithms and simulated annealing to explore an environment in search of better objectives (Carely and Svoboda 1996). There is no requirement that agents be homogeneous in a model; multiple classes of agents, each with their own schema, can be introduced into the same model. The interaction of dissimilar populations can be can easily be modeled.

²⁴ Object oriented programming languages provide a native framework for writing code for an agent which can possess its own methods and properties and be easily replicated throughout a program. This, combined with the creation of standardized agent libraries such as the Santa Fe institute's SWARM (Minar, et.al, 1996) opened up this class of models to researchers without expertise in computer science.

Uses in the enterprise

Towards the end of the 1990s, social science researchers began to acknowledge the utility of agent based models for modeling social systems, both for generating and validating theory and gaining insight into specific systems (Anderson 1999). Social scientists saw the potential of agent-based models to help them uncover the micro-to-macro relationship at the foundation of sociology (Schelling 1978; Coleman 1990). Agent-based modeling allows researchers to build models of bottom up social phenomena, such as standing ovations (Miller and Scott 2004), and then analyze these models, varying the rules, to better understand and even guide the emergence of these behaviors.

To many social science researchers, agent based models represented a “third way” of theory development, in addition to deduction and induction (Axelrod and Tesfatsion 2006). The field of computational organizational theory, described in Chapter 2, was developed in large part in reaction to the introduction of agent based models to social science researchers who hoped to study complexity in social organizations (Carley 2002).

Carley notes that agent based models have been applied in the social sciences along two different paths: an intellectual path, where models are used to generate and test theories, and an emulative path, where models are used to emulate real world organizations with complex dynamics to support change management efforts. Intellectual models tend to use extremely simple agents in highly stylized settings, while emulative models use much more realistic agents and operating environments. While there are obvious differences between these paths, she argues that these models lie on a continuum, and that emulative models such as Levitt's Virtual Design Team (Levitt 1994) have been used to test elements of theory, and intellectual models such as ORGAHEAD (Carley and Svoboda 1996) have been used to suggest specific adaptation policies within corporations (Carley 2002).

Strengths

The primary strengths of agent based models are their ability to model bottom up behaviors in systems with large numbers of locally directed entities, and their flexibility and adaptability. Agent based models are the only simulation methodology that allows the modeler to capture bottom up dynamics driven by the interactions of autonomous agents, such as people in organizational settings. Individual agents can possess memory, environmental awareness, and artificial intelligence. The agents in these simulations may be modeled with high fidelity, relaxing rigid assumptions that must be made when modeling using mathematically based approaches. For example, most equation-based models of economic systems assume the presence of rational actors (the so-called *homo economicus*), despite the fact that economists know that such an assumption is not realistic. The assumptions of rational actors can be relaxed when using agent based models, drawing upon cognitive science to offer more realistic representations of the internal processes actors employ during decision making (Sawyer 2003)²⁵.

Agent based models are capable of modeling a wide array of problems in different contexts. They can be used to model systems using either homogeneous or heterogeneous entities, depending on the fidelity required for the problem. Populations of agents with different characteristics may interact within the same model. For example, a simulation of communication and knowledge in an organization might employ two different populations: one consisting of agents who routinely contact only a handful of others, with another population of agents that travel widely through the organization making contacts.

Agent based models can also be designed such that they evolve and adapt their structure over the course of the model's execution. Agents can be used to explore a problem space, developing structures through exploration using

²⁵ The field of Agent-based Computational Economics explores economic theory using boundedly rational agents in agent based models. See (Tesfatsion, 2002).

schemas that incorporate heuristic search algorithms such as simulated annealing or genetic algorithms that employ crossover and mutation (Carley and Svoboda 1996). Such exploratory simulations have been used both for organizations of people as well as for the development of smart supply chains (Scheritz and Größler 2003). This capability allows modelers to create simulations without knowing the macro-structure of the system *a priori*, instead relying on the micro-foundations of the agents to cause the macro-structure to emerge.

Agent based models have also proven popular because their structure is often simple to convey to model users (even though the behavior of the model may not be simple). They often lend themselves well to use in a game-like environment, encouraging users to test assumptions, vary parameters and rules, and test different scenarios (Guyot 2006).

Weaknesses

While agent based models have proven extremely popular for researchers and business users alike, they do have weaknesses. Perhaps the largest weakness is that the behavior of the models can be very difficult to validate or verify. Unlike equation-based methods such as system dynamics or even Petri Nets, there is no “best” way to verify or validate the relationship between agent’s micro schema and the macro-behavior of the system that emerges. Kulik (2006) described the “indecipherable and seemingly nonsensical analysis” that is common in agent based models, especially when viewed by traditional organizational theorists. He argues that many agent-based models do not have a sufficient grounding in organizational theory to make validation against extant theory possible. Louie and Carley (2008) attempt to balance such criticisms, especially for use in theory development, by claiming that while agent-based models do have drawbacks from a theoretical perspective, they remain the best available tool for studying complex systems when (1) the linkage between micro- and macro- in a system is not well understood, and (2) obtaining information from real world systems is

prohibitively expensive or risky. Even for more emulative models that do not face as much theoretical rigor, it may be difficult to validate that the schema employed in the model does an appropriate job of modeling real-world behaviors and incentives, as the relationship between micro- and macro- is not well understood. Further, performing sensitivity analysis on many agent-based models can be computationally prohibitive (Rahmandad and Sterman 2004). In place of formal verification and validation, several authors have proposed alternative approaches to testing of agent-based models for reliability and usefulness (Miller 1998; Balci 2003; Louie and Carley 2008).

A second weakness of agent-based models is that although they can be applied to a wide range of problems, there are many classes of problems in the enterprise for which they are not suited. Not all problems exhibit bottom up behavior; many aspects of enterprises are indeed controlled in a top down fashion, such as processes. Many aspects of the enterprise are routine and deterministic, and may be better modeled (i.e., with less effort or with more explanatory power) with discrete event simulation or system dynamics.

Another weakness of agent-based models is that they can become very complicated and computationally intensive, especially compared to other simulation techniques. There is a tradeoff to be made between a model's complexity and the ability of the modeler to understand and build confidence in the model. As the number of heterogeneous agents in a model is increased, the number of parameters in the model must also be increased, complicating model testing and evaluation (Rahmandad and Sterman 2004). Large numbers of agents, with complex schemas that incorporate artificial intelligence and memory, can consume significant computational resources.

Agent based modeling is a very powerful methodology with an active and growing user base. Its capabilities and flexibility make it a "go to" simulation methodology for many classes of problems. Unfortunately, however, it is

incapable of modeling all aspects of enterprises. It may not be appropriate to create a agent based model of an enterprise's processes or computer systems; likewise, strategic planning processes or resource allocation might not be easily captured using an agent based approach.

Summary of Methods

Each of the three simulation methodologies presented—discrete event simulation, system dynamics, and agent based modeling—offer the power to capture some dimension of an enterprise's behavior, as it relates to its structure. No one methodology is sufficient to capture every facet of an enterprise's behavior, spanning all of the "views" employed in an enterprise's architecture. An enterprise modeler must then select a methodology that best fits given a particular problem at hand, based on an understanding of each methodologies strengths and weaknesses.

Table 3-2: A comparison of potential simulation methodologies

	Perspective	Audience	Computational Logic	Unit of Analysis	Strengths	Weaknesses
Discrete Event Simulation	Top Down	Process owners; operations, architects	<ul style="list-style-type: none"> • Discrete time • State transitions • Entity flow • Stochastic 	Entity	<ul style="list-style-type: none"> • Easy to understand, highly representative • Stochastic; timed • Straightforward to calibrated/ validated to a specific process/structure • Lends itself to detailed statistical analysis 	<ul style="list-style-type: none"> • Can only model systems from an entity flow perspective • Entities have no intelligence
System Dynamics (Differential Equation)	Top Down	Management, architects	<ul style="list-style-type: none"> • Continuous time • Differential equations • Causal structures & variables 	Stocks and flows	<ul style="list-style-type: none"> • Relates behavior to causal structures and feedback loops • Captures macro-level observable variables • Can be used with qualitative variables • Relatively easy to implement • Ease of analysis of underlying mechanism 	<ul style="list-style-type: none"> • Treats resources within the model as homogeneous, continuous • Sensitive to initial conditions, missing causal loops
Agent based Model	Bottom Up	Management, architects	<ul style="list-style-type: none"> • Discrete time • Interaction of heterogeneous agents with local schemas 	Agent	<ul style="list-style-type: none"> • Can simulate bottom up behavior driven by localized action • Can model heterogeneous agents • Can model structure as it emerges from localized behavior • Very flexible 	<ul style="list-style-type: none"> • Difficult to verify and validate • Large computational power required

3.4.4 Comparison of Simulation Methodologies for Enterprise Architecting

Table 3-2 presents an overview of the three simulation methodologies presented in this section, comparing their perspective, computational logic, audience, strengths and weaknesses. When selecting which simulation methodology to use to model a problem, the modeler should consider the relative strengths and weaknesses of potential methodologies, including such factors as analytical capability, ease of communication to model users/stakeholders, and difficulty of implementation. The modeler should heed of the adage “when all you have is a hammer, the whole world is a nail:” if a modeler is only familiar with one or two simulation methodologies, they were see solutions employing their preferred tool even when an alternative methodology may be superior. While for some behaviors, such as organizational dynamics driven by heterogeneous motives with a micro-to-macro relationship, there is only a single choice of modeling methodologies, many issues could be modeled with multiple simulation methodologies.

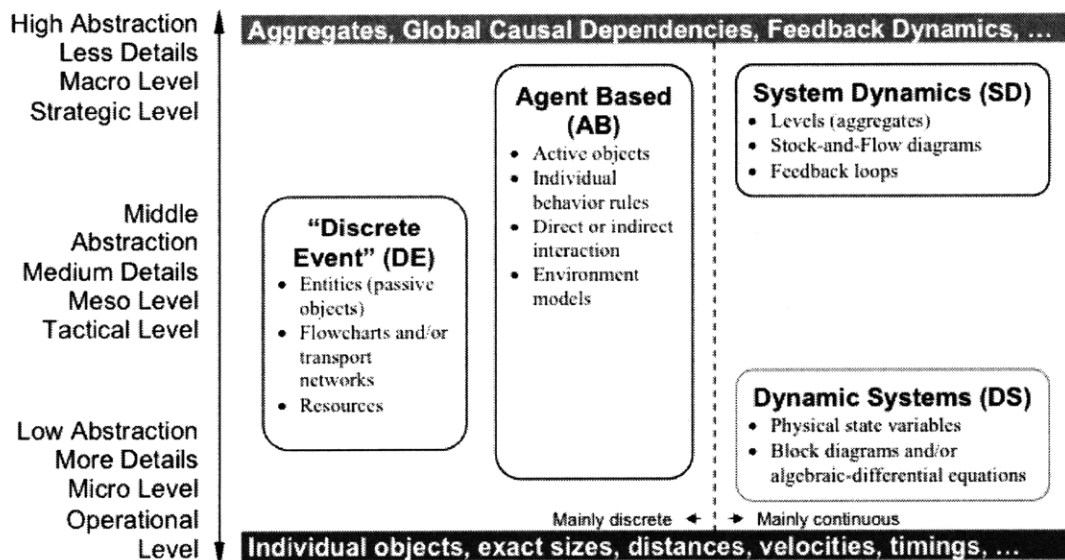


Figure 3-5: Application of Simulation Methodologies on Abstraction Level scales.
 From (Borshchev and Filippov 2004).

Figure 3-5, from Borshchev and Filippov (2004), provides an approach to visualizing where these simulation models are strongest. This figure shows the modeling space ranging from micro-to-macro levels of system abstraction and divided between discrete and continuous representations of systems (dynamic systems, show in the lower right hand corner, are micro level simulations of physical systems, as a finite element analysis simulation used by mechanical engineers). The figure shows macro-levels of abstraction corresponding to strategic models, with mid- and micro-levels of abstraction corresponding to tactical and operational modeling, respectively.

Surveying the literature and in practice, discrete event simulation is overwhelmingly the preferred approach for modeling processes which do not require intelligent agents. They are capable of modeling mid-level to micro-level behaviors found in processes, often employing highly developed and specialized graphical tools for process modeling. Processes could possibly be modeled with agent based models, but doing so would require significant modeling effort when compared to a discrete event model without providing any additional benefit. For micro-level modeling of systems with micro- to macro- behaviors, or for systems that require adaptation of the model's structure during runtime, agent based models are usually the preferred approach for modeling the system behaviors.

For many macro-level systems, however, the choice of modeling approach is less clear; both discrete, intelligent agent based models, and continuous, equation-based system dynamics may provide viable approaches. Rahmandad and Sterman argue that agent based models and system dynamics are best viewed "as points on a spectrum of aggregation assumptions rather than as fundamentally incompatible modeling paradigms" (2004: 3). Indeed, it is possible to model many systems with either approach. Rahmandad and Sterman compared both agent based models and system dynamics models of disease contagion, based upon the standard SEIR model for disease propagation(2004).

They found that for most configurations of the models, the outputs were very similar, despite including various levels of heterogeneous agents with varying social network structures (e.g., random, small world, scale free) in the agent based models. They concluded that extensive disaggregation possible with agent-based models may not be warranted unless detailed data characterizing the micro-interactions is available, the structure is stable, and the computational burden does not prevent sensitivity analysis.

Bobashev et al. (2007) performed a similar analysis comparing agent-based modeling and system dynamics for models of disease contagion, finding that different models excelled at capturing different phases of contagion, and proposing a joint, hybrid model be used to model the process in a more computationally efficient and analyzable way. In both Rahmandad and Sterman's study, as well as Bobahsev's study, it was determined that while either methodology might deliver similar output, the operation of the model may reveal different insights into the system's behavior. In some cases, it may be conceptually clearer to think of the problem in discrete terms, such as the travel of specific disease carriers through a population of uninfected people. In other cases, this paradigm might not make sense, and a homogeneous modeling approach may offer similar information while providing more powerful analysis tools. This might be the case when modeling financial systems, where units of currency are homogeneous and do not benefit from an intelligent, heterogeneous modeling approach capable of tracking the history of individual units of currency.

While there are no tests that can specifically determine the optimal simulation methodology to use when modeling a specific problem and its dynamics, the preceding discussion can be used as guidelines for choosing a simulation approach that provides insight into the dynamics of the problem and is feasible to implement, in addition to just providing the correct analytical output.

3.5 HYBRID SIMULATION MODELING

Most complex socio-technical systems, such as enterprises, have many different behaviors with different attributes: macro-level, micro-level, micro-to-macro level, discrete, and continuous. The views used by enterprise architecture frameworks decompose the enterprise into a set of views that correspond to the different contexts of enterprise behavior. Each view has its own unique behaviors, driven from the top down or bottom up, from populations of intelligent, locally aware actors, or from directed, stable, repeatable processes. While the views could be simulated individually using whatever simulation approach best fits the context of that view, there would be no way to simulate how the enterprise architecture behaves as a system. The enterprise architecture is a near-decomposition of the enterprise, in the sense of Simon's use of the term. While the views are tightly coupled internally, there remain other, "loose" couplings between the views. If the enterprise architecture were to be simulated as a system, complete with the interaction of its loosely coupled views, the various simulations of the views must be connected together to form a hybrid simulation of the enterprise architecture.

To date, there has not been a concerted effort within the field of enterprise architecting (or computational organizational theory) to use models to understand the dynamic relationship among architectural views. The ARIS enterprise architecture framework and methodology employs Petri Nets to model processes; other frameworks employ UML or IDEF models to describe the static relationship between components within and view, but none of these frameworks use any form of modeling or simulation that extends across the views of the enterprise, despite the fact that all enterprise architecting frameworks recognize the importance of interactions across the views of the enterprise.

While most enterprise architecture frameworks use a set of views to capture various aspects of the enterprise, *there has not been an attempt by any enterprise architecture framework or toolset to employ models that explore the linkages among these views.* This is in part because the views have very

different contexts; each view uses a different type of analysis to understand the enterprise from different perspectives. For example, the thinking and analysis used to describe how individuals in organizations are incentivized to behave is very different from the thinking and analysis used to describe how strategy is implemented in response to changes in a competitive, regulated marketplace.

The reason as to why there currently exists no single simulation methodology that can capture the enterprise's dynamics holistically is because each simulation methodology has its own perspective and biases and that perspective does not necessarily do the best job of capturing the specific context of every architectural view. In order to best capture the specific context of each view, each view must be simulated using a methodology that is best suited to its particular context. These views are not independent, however, and must be given the capability to inform each other. The various simulations of the views can be linked together via variables that exist across two or more of the views in a way that allows the outputs of one view's simulation to inform the behavior of a second. In such a structure, each view's simulation would be considered a sub-model, and the sub-models would be integrated to form an enterprise model includes simulations of multiple views. In this way, feedback loops existing among architectural views and their contexts can be analyzed and understood. By using the multi-methodology approach described above, the modeler can create a hybrid simulation that does a better job of capturing the dynamics of the entire system than a single methodology model could.

3.5.1 Background of Hybrid Enterprise Simulations

The idea of hybrid simulation modeling is not new. For decades, the idea of linking together simulations that capture different contexts of the same system has been used in fields ranging from physics and biology to aerospace engineering. These hybrid approaches use the outputs of one sub-model as the input into another. Mingers and Gill (1997) were the first to suggest using multiple

modeling methodologies in the social and management sciences to understand firms from an organizational science perspective, espousing the use of the soft systems approach (Checkland 1981) in conjunction with other management science modeling methodologies as a holistic approach to modeling and understanding management strategies. They were the first to use the concept of differing *contexts* in the enterprise to motivate the use of different modeling methodologies. Their approach did not employ dynamic simulation, per se, but did open the door to the idea of employing different simulation methodologies linked together in a single, hybrid simulation model to help understand complex socio-technical systems that exist across multiple contexts.

More recently, with advances in available simulation software, there have been more efforts to use a hybrid approach for simulating the enterprise, often mixing a macro-level, homogeneous model employing system dynamics with an agent-based, heterogeneous micro-level model in order to model the system from both top-down aggregate and bottom-up disaggregate contexts. There have been several calls to study the complementary features of these two methodologies for the purposes of creating hybrid models that capture emergent behavior arising from the structure of systems (Scholl 2001; Borschsev and Phillipov 2004; Rahmandad and Sterman 2004).

The first such hybrid system dynamics/agent-based models were tactically focused on understanding the emergent structure and dynamics of supply chains (Schieritz and Größler 2003). Supply chains and production planning have proven to be an excellent test bed for hybrid modeling. These topics have received much attention from the modeling community both because of their importance and because of their fairly well defined boundaries, inputs, and outputs. Both supply chains and production planning have traditionally been simulated using discrete-event models of their process flow, connections, inventories, and timing. Recent hybrid simulations of supply chains have married the micro-level perspective of discrete event simulation at a tactical level with the

macro-level perspective of system dynamics to capture industrial and market dynamics (Rabelo et al., 2007). Production planning has been modeled in a similar way, with micro-level models of the process (discrete-event) informing macro-levels of strategic planning (system dynamics) (Rabelo, et. al. 2005; Venkateswaran and Son, 2005).

3.6 CONCLUSIONS

This chapter has explored the literature related to modeling enterprises and enterprise architecture. First, current approaches to modeling enterprise architecture, such as IDEF and UML were explored and were found to lack the ability to help enterprise leaders understand the behavior of their enterprise. This was used to then explore the needs of enterprise leaders with respect to simulation models, where a list of criteria was developed for a simulation methodology to be useful. With these points in mind, three simulation methodologies discussed: discrete event simulation, system dynamics, and agent based modeling. A brief overview of each approach was given, along with a discussion of each methodology's strengths and weaknesses, and a general comparison of the application of each. Because each approach is best applied in a specific context, and many complex systems are composed of multiple contexts, hybrid simulation models were proposed as a potential way to simulate complex systems as a holistic system without sacrificing multiple perspectives of the system each with their own context, and potentially their own sub-model. The application of hybrid modeling to the enterprise was then reviewed, and it was shown that while very useful, most applications to date have been limited in scope, and have not been able to look at larger enterprise issues.

This paves the way for the development of an approach to hybrid simulation modeling of enterprises that is able to capture the enterprise as a system, linking together multiple perspectives of the enterprise using its architecture. The following chapter will develop principles for developing hybrid simulation models,

and propose a process that can be used to create these hybrid models of an enterprise's architecture for the purposes of aiding high level management.

Chapter 4: SIMULATING ENTERPRISE ARCHITECTURE USING A HYBRID APPROACH

This chapter presents an approach aimed at addressing the modeling needs of enterprise architects and managers as they guide the development and evolution of their enterprises. It begins by defining a set of guiding principles that can be used to develop a hybrid architecture-based approach to enterprise architecture modeling. This chapter explains how enterprise architecture frameworks, in combination with hybrid modeling techniques, can be used as a basis for the creation of such simulation models. The bulk of this chapter will outline a process for using this simulation technique to support enterprise architecting by focusing on a salient strategic issue or question facing top enterprise leadership, by considering key associated technical issues. Finally, this chapter concludes by examining how architects and managers can use this approach to inform critical decisions in the management and development of their enterprises.

4.1 PRINCIPLES FOR CREATING HYBRID SIMULATION MODELS OF THE ENTERPRISE ARCHITECTURE

Simulation models have been used for a wide range of applications, including forecasting, education, theory-building, and as decision aids. Each application requires a different approach to model building and use, as each application has different objectives and goals. A simulation model intended predict the price of a traded commodity, for example, requires a different approach to create and use than a simulation model intended to understand the spread of a contagious disease, or one used to teach people how to use a system. The guidelines used to create a simulation model change depending on whether the model is intended to predict, optimize, build theory, teach, communicate, or uncover hidden mechanisms.

The goals of simulation models espoused by this thesis are focused on understanding and communicating the effects of the enterprise's architecture on the behavior of the enterprise, with enterprise leaders as the key model users. These simulation models must yield insight into complexity, while communicating and educating. They do not need to be predictive, but they must be able to identify possible enterprise behavioral and performance outcomes in response to discrete strategic management decisions, given its defined architecture. These simulation models must work well with existing abstractions, and integrate disparate perspectives into a single hybrid simulation model. To meet these goals, a set of guiding principles have been developed to help guide the modeling process. There are four key principles that can be identified to aid the modeler in creating hybrid simulation models of an enterprise architecture:

1. Models should be created for insight (e.g., through "what if" analysis, by defining possible future outcomes), not for generating point predictions;
2. Models must capture the essential elements of the enterprise's architecture;

3. Hybrid models should be preferably focused at the strategic level, not at the tactical level to address enterprise-level strategic decisions and their possible consequences;
4. Hybrid models must explicitly capture interactions across the enterprise's architecture, comprising multiple views or domains, as required by the strategic decision question or issue being posed.

The next four sections develop these principles for application to the creation of hybrid simulation models of enterprise architecture.

4.1.1 Modeling for insight, not prediction

Enterprise architecture simulation modeling is not a *predictive* approach to simulation modeling intended to improve operational efficiency (as most are); instead, this approach seeks to yield *insight* into the behavior of a complex enterprise arising from its *architecture*. This *insight*, in turn, helps to accelerate the learning curve for enterprise managers seeking to shape and guide their enterprise from a system-level perspective. This approach allows the modeler to capture key attributes of the enterprise from multiple perspectives (*e.g.*, strategy, process, organization, products, etc.) and to examine how the interactions between these perspectives drive the high-level behaviors of the enterprise. Such an approach takes a strategic view of the enterprise to guide enterprise architects and managers in understanding how the system, as a whole, delivers value to its stakeholders.

This use of simulation modeling as a strategic decision support tool is similar to the position espoused by researchers in the fields of systems thinking and organizational learning. Here, models are used to build insight into the system by:

1. Creating a shared frame of reference among managers and architects for understanding non-linear enterprise dynamics;
2. Understanding the relationship and effects between both "hard" (quantitative) and "soft" (qualitative) system variables;

3. Testing hypotheses and performing scenario analyses; and
4. Discovering new architectures or ways to manage the current architecture as a result of analysis of the simulation.

The last three points correspond closely with the three stages of learning espoused by researchers in the field of organizational learning (Argyris and Schön, 1978; Senge, 1990; Sterman, 2000; Fowler, 2003).

4.1.2 Modeling the architecture, not the enterprise

This approach models the *enterprise architecture*, as opposed to the enterprise itself. As defined in Chapter 2, the enterprise architecture is an *abstraction* of the real-world enterprise that captures the essential policies, structures and processes of the enterprise that allow it to provide value to its stakeholders; it is the high-level mapping of the enterprise's function to its form. It is not a complete enumeration of every linkage, structure and policy of the enterprise, nor does it include the contributions and capabilities of individuals.

The real-world enterprise is both complicated and complex. Most enterprises are in a constant state of flux, filled with locally aware and autonomous people and many ad hoc structures constantly changing and adapting to their environment. It is practically impossible to accurately predict the future behavior of such a complex system. To accurately model the real-world enterprise would be analogous to creating a model to predict the image from a common kaleidoscope. Any detailed model of a kaleidoscope will be wrong because, although some parts are fixed, most kaleidoscopes contain loose, tumbling colored beads, sequins, and bobbles that move in a random fashion, which can't realistically be modeled.

In contrast to the real-world enterprise, the enterprise architecture is more stable, changing only to realign with major shifts in the environment or in long-term strategy. By modeling the architecture instead of the enterprise itself, it is

possible to achieve a deeper understanding of how the enterprise behaves over a wider range of inputs. To come back to the kaleidoscope analogy, modeling the enterprise architecture is similar to understanding that a kaleidoscope is a tube containing of a number of mirrors set at angles with loose, colored objects that produce spectacular images when viewed. With just this understanding, the “modeler” won’t be able to reproduce an arbitrary image or even make an accurate prediction as to what the next image will be at some future point in time, due to the randomness of the colored objects. However, the “modeler” will be able to understand the effect on the image of adding more mirrors, changing their angles, or adding different kinds and colors of loose objects. This kind of knowledge about the kaleidoscope’s architecture is much more practically useful than a detailed predictive physical model of the movements of beads and sequins in a chamber.

4.1.3 Strategic Level Modeling of the Enterprise Architecture

A key tenet of the proposed approach to hybrid modeling of enterprise architecture is the focus on enterprise dynamics at a strategic level, as opposed to tactical or operational levels. This has profound implications for the modeling approach: instead of a detailed model with high precision that aims for predictive capability in a single context, this approach aims to deliver a model that, while lacking high precision, delivers insight about the system in a far broader context in the face of uncertainty. Such an approach to modeling addresses fundamentally different questions from those addressed by tactical models. While a tactical model might answer the question “what are the parameters that will allow me to achieve a optimum output?” a strategic model will answer “what are the design characteristics that provide for good performance in my environment?”

While organizational design theorists, systems thinkers and enterprise architects (Galbraith 1973; Sterman 2000; Ross, et al. 2006) have embraced a strategic focus in the study of enterprises, this has not been reflected in the work of

enterprise modelers. Most modelers tend to focus models either too narrowly (such as to support a single view in an enterprise architecture), too tactically (as in the study of supply chain efficiency) or both (Kalpic and Bernus, 2002; Epstein, 2003; Schieritz and Größler, 2003). Researchers in the field of computational organizational design have created models with a strategic focus (Rivkin and Siggilekow, 2003; Siggilekow and Levinthal, 2003; Ethiraj and Levinthal, 2004), but these models tend to be theory generative and too abstract to provide insight that most enterprise managers can understand and trust. There is a tremendous need for a strategy-focused modeling capability that will not only support the theories of organizational design theorists and systems theorists, but will also support the needs of enterprise managers as they seek to guide the development and direction of their enterprise.

4.1.4 Focus on dynamics resulting from by interactions across the architecture

The proposed approach to simulation modeling of enterprise architecture seeks to capture the complex dynamics of the enterprise by explicitly capturing the interactions across the contextual boundaries of the enterprise architecture's views, in accordance with the concept of near-decomposability of complex systems. These interactions provide pathways for feedback in the system, both within and across the views of the enterprise architecture. Beginning with a set of initial conditions, such a simulation model will play out the response of the enterprise's architecture over time to an external environment.

4.2 A PROPOSED HYBRID, ENTERPRISE ARCHITECTURE FRAMEWORK-BASED MODELING APPROACH

This thesis proposes a hybrid approach to simulation modeling of enterprise

behavior employing enterprise architecture frameworks. With this approach, the needs of today's enterprise leaders are met for a representative, timely, adaptable approach to modeling that yields insight into the underlying behavioral complexity of the enterprise. This approach is particularly useful for modeling those enterprise behaviors and functions that cross multiple views of the enterprise architecture and cannot be captured fully by any single simulation methodology.

This new approach addresses both the technical methodology for creating the simulation model as well as the process required to (1) develop an appropriate question for the model to answer, (2) gather data, (3) create the model, and (4) learn from its application. This approach is uniquely suited to meet the needs and demands of enterprise architects and managers by providing them with a holistic and strategic perspective of their enterprise that allows them to ask targeted questions to uncover the dynamics that drive their enterprise's *architecture*.

There are two key characteristics of the approach to simulation modeling of enterprises set forth in this thesis:

(1) a hybridization of simulation techniques (such as system dynamics models, agent-based models, and discrete-event models) that connects the inputs and outputs of multiple sub-models, and

(2) the use of an enterprise architecture framework to provide a grounded abstraction of the enterprise and identify the sub-models as well as the boundaries and relationships between sub-models.

This approach is employed to gain insight into how the interactions across the views of an enterprise architecture drive the behavior of the enterprise. Figure 4-1 presents a simple representation of the high-level structure of a hybrid simulation approach.

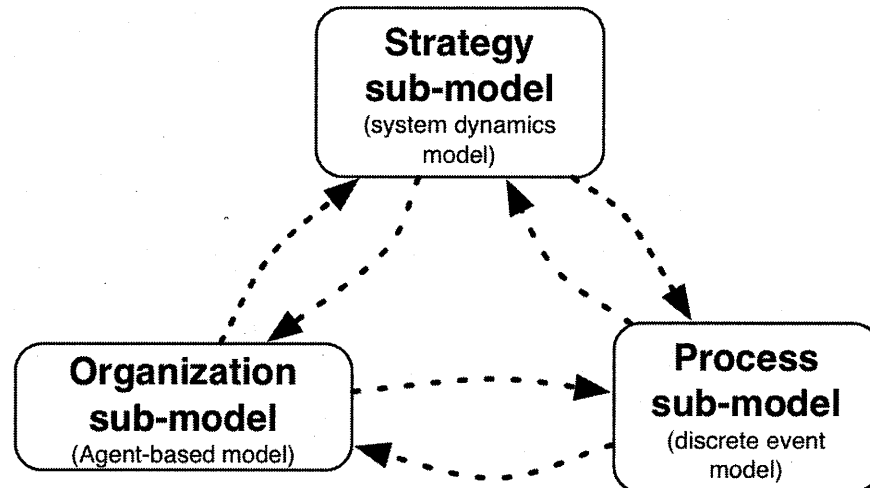


Figure 4-1: A notional representation of a hybrid, enterprise-architecture based simulation model

Figure 4-1 depicts the basic form of a hybrid model composed of three sub-models: strategy, process, and organization. Each of these sub-models corresponds to a specific view in the enterprise architecture framework that the modeler has chosen to use. For the sake of presentation, only three views are shown, but more can be used. Each sub-model/view is modeled using a simulation methodology that best captures its particular dynamic *context* (e.g., top-down, bottom-up, macro-scale, micro-scale, discrete time, continuous time, agent interaction, stocks and flows, process timing, etc.).

The sub-models are linked together through a set of shared variables or events that serve as interfaces, shown as dotted lines. In every case, one of the sub-models acts as a controller for the interface variable, representing that sub-model's output, and determines its value, which is then used as an input by another sub-model. There can be multiple inputs and outputs connecting the sub-models. An effective enterprise architecture modeling framework that supports this hybrid approach will help define not only the boundaries between the sub-models via the use of views, but will also characterize the classes of interactions among the views. A framework allows the modeler to develop sub-models consistent with a set of architectural views while explicitly linking the

behavior of these views together via, for example, feedback-looped relationships. In this way, enterprise architects and managers alike can understand and relate to these resulting interaction effects among the various enterprise views from an organization or enterprise-level strategic systems perspective.

While a simulation model in and of itself is an interesting and useful tool, its impact is greatly augmented when it is used as part of a coordinated architecting process. This iterative process includes (1) framing a question to guide model development, (2) scoping the model, (3) gathering information for the model, (4) developing the model based on this information, and (5) testing and evaluating the model to build confidence in its usefulness and explore its implications, and (6) use the model by conducting simulation experiments to help define the future-state enterprise architecture options as part of the enterprise architecting process. The next section will formally develop this process.

4.3 A PROCESS FOR DEVELOPING HYBRID MODELS OF THE ENTERPRISE





Simulation models are used as part of an iterative *process* to help understand problems with the current architecture and provide input for creating new “to-be” architectures. This process helps ensure that the simulation model is properly bounded and scoped, that a framing question is well posed, that the model is created in a logical progression, and that it can indeed be useful in answering the question it was intended to answer. While modeling will always be a creative endeavor, it can benefit from a structured process. Without a well developed process to guide the model creation effort, the model can quickly become unmanageable or unsuitable to its purpose as modelers lose sight of model scoping, purpose, or structure. A process that is standardized helps to ensure that modelers do not lose sight of the end goals and structure the model in ways that make it more useful to decision makers and for later use. The process

serves as a tool to guide the creative process in much the same way that an enterprise architecture framework is used to create an enterprise architecture.

The process of creating a model is not a strictly linear process. All modeling is iterative, with updates to the model as more information becomes available and new insights are made. As the model is created, it should be continuously tested and reevaluated to ensure that it is meeting its objectives. While testing is often only undertaken towards the end of the modeling process, there is no reason as to why preliminary tests cannot be performed during model development. After the model is evaluated, new hypotheses can be developed, and the process can be iterated again.

The general form of the process outlined in Table 4-2 is adapted from Sterman's process for modeling business dynamics (2000, Chapter 3), amended to support a hybrid, enterprise architecture based approach. Sterman's approach is flexible, and oriented towards the development of multi-part system dynamics models of businesses, making it easy to adapt for use to create hybrid models by emphasizing boundary setting steps and adding in steps specific to hybrid modeling, such as simulation method construction, and sub-model integration. Each of these steps will be developed in further detail in the following sections.

Table 4-1: Steps of the hybrid, EA-based simulation modeling process. Arrows indicate major feedback loops. Adapted from Sterman (2000).

	<p>1. Document the enterprise architecture</p> <ul style="list-style-type: none"> • Use an EA framework that supports dynamic modeling, defining views and heir interactions
	<p>2. Problem articulation</p> <ul style="list-style-type: none"> • Is this the proper approach to answer this problem? • Identify: <ul style="list-style-type: none"> • Problem, Key Variables, Critical Behaviors, Time Horizon
	<p>3. Form a Dynamic Architectural Hypothesis</p> <ul style="list-style-type: none"> • Initial hypothesis generation • Endogenous focus
	<p>4. Identify the applicable architectural views</p> <ul style="list-style-type: none"> • Downselect EA views based on problem dynamics <p>5. Match views with simulation methodologies</p> <ul style="list-style-type: none"> • Match based on context of dynamics, required inputs/outputs • Select a modeling environment to be used <p>6. Identify boundaries and interfaces among view sub-models</p> <ul style="list-style-type: none"> • Use EA to identify boundaries and interfaces relevant to the problem and dynamic hypothesis • Create sub-model boundary charts <p>7. Create sub-model and top-level model diagrams</p> <ul style="list-style-type: none"> • Create sub-model diagrams of the structure of each sub-model • Create top-level model diagram that depicts how the sub-models will be linked to create the top-level model. <p>8. Create the Simulation Model</p> <ul style="list-style-type: none"> • Estimation of variables, relationships, and initial conditions • Model sub-views with selected methodology • Combine sub-models into an architecture model • Develop global model interface <p>9. Model Testing</p> <ul style="list-style-type: none"> • Sensitivity Analysis • Behavioral and Structural Analyses • Other analysis methods (see Table 4-3) <p>10. Policy Design and Evaluation</p> <ul style="list-style-type: none"> • Scenario Analysis • "What if" Analysis

4.3.1 Step 1: Document the Enterprise Architecture

Modeling enterprise architecture dynamics with this approach requires that there is a documented enterprise architecture (or one under development) that can support a holistic hybrid modeling effort, in keeping with the previous descriptions of the requirements for defining a useful enterprise architecture for modeling purposes. Ideally, the enterprise architecture is something that is created and maintained independently as the enterprise evolves and adapts and can be used as a guide for the creation of the model, rather than being something that is created specifically to meet the needs of the model. If not, then creating/documenting the enterprise architecture *de novo* would be a significant undertaking. This initial "step" of the process should be seen as a necessary condition rather than a discrete step that is repeated for the creation of a new model.

Unfortunately, many of the existing enterprise architecture frameworks do not have the structure needed to facilitate the creation of hybrid simulation models. An appropriate framework will not only define a set of views for decomposing the enterprise, but will also specifically define the interfaces between or among these views. The views used must span the structure and operation of the enterprise and its environment, with a scope similar to that of the CEO, rather than place emphasis on a single aspect of enterprise operations, such as processes or information systems. While enterprise architecture frameworks are continuing to mature in this direction, many of the most popular frameworks, such as the Zachman Framework or TOGAF, do not meet these requirements without modification. The Nightingale and Rhodes Enterprise Architecture Framework, currently under development, is one such framework that has sought to decompose the enterprise from the perspective of the CEO, while capturing the high-level interactions among the views. The Federal Enterprise Architecture Framework is also rapidly moving towards supporting such a capability. As enterprise architecture continues its trend to be used by higher level enterprise

leaders, these capabilities will become more common in enterprise architecture frameworks.

When an enterprise architecture is actively used and kept current, the time required to create a simulation model is drastically reduced, and it becomes realistic to assume that model creation can become timely enough to be useful to enterprise leaders. If an enterprise's architecture is not documented, the work of creating one from scratch can consume many months, significantly delaying the creation of the model and reducing its eventual utility.

4.3.2 Step 2: Problem Articulation

Enterprise leaders and architects will often begin with a general question that they hope simulation modeling might be able to help them answer. This general question is often vague, lacks structure, may address a symptom rather than the root problem, and may be difficult to concretely answer. It likely does not identify key metrics, boundaries, or conditions for the model, and is potentially open-ended. Without further developed guidance and purpose, a simulation model built to answer a vague, general question can quickly grow beyond its necessary scope, incorporating needless detail and increasing in complexity until it becomes unmanageable and ultimately not useful.

The general question should be developed into a problem statement to provide specific objectives and conditions that the model must meet in order to address the problem. Wherever possible, it should specify the particular architectural structures that are under consideration (incentive structure of executive management vs. "organization"), the metrics that will be used as inputs and outputs to the model (ROIC vs. "performance"), the time horizon (two years vs.

undefined) and the environment (a list of scenarios vs. “a changing environment”).

As the problem is articulated, the modeler must ask if a hybrid simulation approach is necessary. For a hybrid simulation to be potentially useful, the problem must be *dynamic* and arise due to the *architecture* of the enterprise. A simulation of the architecture will not be helpful to model demand, nor should it be used to model the individual decision characteristics of individuals. The problem must be tied to the architecture. Further, it must be dynamic: what will the performance of the enterprise be over time? Finally, the problem must cross multiple views of the enterprise if a hybrid approach is to be used. Most tactical level problems, such as process improvement activities, can be modeled entirely from within a single perspective using a single simulation approach, greatly saving time and resources without sacrificing its analytical capability. Hybrid simulation models are typically needed for only high-level, truly enterprise-wide problems that span multiple views. As the problem is articulated, it is critical to appropriately characterize it so that the right modeling or other analysis approach is chosen.

Problem articulation is not easy. It will likely require several iterations with stakeholders to develop a problem statement that both addresses the problem at hand and can serve as a guide for the creation of the model. It is, however, worth spending this time developing the problem statement before embarking on a modeling safari that will require significant rework after the problem finally becomes clear.

4.3.3 Form a dynamic architectural hypothesis

After clearly defining the problem, the next step is to develop a *dynamic architectural hypothesis*²⁶. This hypothesis uses the structure and dynamics of the architecture to explain the problem. It is always provisional, subject to revision or abandonment. The hypothesis guides and focuses the development of the model. The dynamic architectural hypothesis must focus on a dynamic, architectural hypothesis to explain the problem at hand. If it does not, then a simulation model based on the enterprise architecture is not the appropriate tool to investigate the problem and hypothesis, and an alternative modeling approach should be used.

4.3.4 Step 4: Identify the applicable architectural views

While the enterprise architecture contains a set of views that span the enterprise, the problem that the model seeks to address does not necessarily require every view to be explicitly captured in the enterprise architecture modeling effort. This step of the process requires the modeler to critically consider the dynamics and architecture of the problem that he or she is addressing with the model.

Within the study of near-decomposable systems featuring interactions with quite often nonlinear feedback, it is not always obvious what is contributing to the overall enterprise's dynamic behavior. For instance, is it possible that organizational incentives affect process flow? Initially, the answer to this question may not be clear, or the connection may be through an intervening step. For this reason, it is best to begin by considering all of the views of the enterprise architecture, and consider the impact of each on the problem the model is addressing. Remove a view only when it can be safely assumed that it does not

²⁶ Sterman calls this the *dynamic hypothesis*. This term has been modified to reflect its use in the hybrid, EA-based approach.

significantly contribute to the dynamics of the problem the model is addressing over the time horizon and set of inputs for which the model will be tested. As work on the model progresses, it may become necessary to revisit these assumptions.

4.3.5 Step 5: Match views with simulation methodologies

After the views to be used have been identified, it is necessary to consider *how* each view will be simulated. The challenge of choosing a simulation method for each view is very similar to the challenge that faces any modeler when beginning a simulation project:

- What approach will best answer the question the model is meant to answer?
- What approach will yield the most insight, or be easiest to implement?
- What is the context of the endogenous dynamics related to the problem *within a particular view*?
- What simulation approach best captures that context?

These questions should be asked when creating each sub-model, directed to the dynamics within the related architecture view. When modeling a sub-model, as opposed to a model of an entire system, an additional question must be asked: how might this sub-model interact with others?

Every simulation approach has its strengths and weaknesses, as highlighted in the previous chapter. The key benefit of hybrid modeling, however, is that fewer compromises must be made—a particular simulation approach is used where it is strongest, and another approach can be used in an area where it is less strong. Weaknesses of one methodology can be offset with the aid of another methodology.

For the purposes of enterprise architecture simulation, each architectural view will require the selection of a simulation methodology that best addresses the

dynamics within that view, *as they relate to the ultimate question being asked of the hybrid model*. The notion of what “best addresses the dynamics” requires the modeler’s best judgment, but it is best achieved by matching the context of the dynamics within a view with capabilities of various modeling methodologies. The context of the dynamics can include a number of different attributes, including the following:

- **Perspective:** top-down or bottom-up;
- **Dynamic Structures:** feedback loops, interacting agents, moving entities/resources;
- **Unit of Analysis:** structure of system or agent’s rules;
- **Level of Modeling:** aggregate variables or highly discretized variables;
- **Structure of Model:** fixed, or dynamic (evolutionary);
- **Handling of Time:** continuous, discrete, event-driven; and
- **Resource Flows:** stocks and flows, agents, entities.

These attributes can then be matched to the strengths of one of the simulation methodologies reviewed in Chapter 3. The intention of this context mapping is to ensure that the simulation approach used is able to bring insight into the dynamics at hand in a straightforward manner.²⁷ Closely related to the question of context matching is the question of effort: how much effort and time is required to capture the relevant dynamics of each view using various methodologies? In most cases, the simulation method that best captures the context will also be the approach that is more straightforward to implement in the particular situation.

²⁷ For example of a non-straightforward application of a simulation methodology, system dynamics can be made to capture the dynamics of a heterogeneous populations by creating numerous instances of stocks and flows, each with their own parameters and perhaps with additional decision logic. While this is technically feasible, it is difficult and convoluted. An agent based model would be simpler, quicker, and likely more insightful in this instance.

Table 3-2 from the previous chapter can serve as a useful guide for comparing the capabilities of these simulation methodologies.

It is important to note that there is no universal mapping of appropriate simulation methodologies onto the list of enterprise architecture views such that one methodology will always be used to model a given view. This is because the dynamics of a given view may change depending on the problem that the hybrid model is called upon to answer. As an example, the organization view used in many enterprise architecture frameworks can be used to frame many different dynamics, such as organizational incentives giving rise to emergent behaviors and decision-making (bottom-up, evolutionary); the alignment of the organizational structure to processes and the product architecture (top down fit and alignment), or cultural changes in response to changes in the environment (feedback system). In this way, the sub-model of an appropriate organizational view could be created with either an agent-based model, a contingency fit model, or a system dynamics model, depending on the context of the dynamics under consideration by the hybrid model.

Although one modeling methodology is being chosen to model each view, these sub-models are not independent. They will interact with other sub-models, and the choice of modeling methodology may potentially depend on some of these interactions. For this reason, this step may need to be iterated with the following step, which identifies the boundaries and interfaces among the sub-models.

4.3.6 Step 6: Identify boundaries and interfaces for each view

After each applicable view from the architecture is identified and matched with a simulation approach, *boundary charts* should be created to help identify the variables, parameters, and boundaries for each sub-model. Boundary charts are intended to help identify model boundaries, separating the endogenous from

the exogenous, as well as what is specifically excluded. These charts are intended to clarify assumptions and boundaries of the model and make review of the models easier. In the case of hybrid model development, they will also be used to explicitly highlight the interfaces between the sub-models of the views.

A model boundary chart, as developed by Sterman (2000), lists endogenous variables, exogenous variables, and excluded variables. Endogenous variables, as the name implies, are variables that are internal to the model itself. There are potentially a great number of these. Exogenous variables are those that are externally imposed on the sub-model and in the case of a hybrid model, these are the inputs from other sub-models. The last factor consists of excluded variables. This list is intended to clearly define what the sub-model will not consider within its scope and helps to avoid future confusion.

For hybrid simulation models, a fourth factor should be considered: outputs. Outputs are a class of endogenous variables that are used as exogenous variables by another sub-model: they are "interface variables" between sub-models. While the identification of endogenous and excluded variables is useful, the identification of the inputs and outputs of each sub-model is critical to the hybrid structure of the model. This defines the interfaces between sub-models and determines the directionality of the interaction between models.

These interface variables link together multiple sub-models by serving as an output from one sub-model and as the input to others. They serve as the direct conduits for the dynamic behavior of the enterprise by making feedback across the contexts of the sub-models possible. While the same variable can have meaning in multiple views, from a modeling perspective, the value of the variable can only be determined by one sub-model. For example, there cannot be three instances of "inventory level," each calculated in a different manner for use in a different sub-model. One sub-model must calculate the value and then export it to other models that may make use of it so that the model stays logically

consistent. This is not to say that there is not feedback that influences the value of interface variables—feedback can occur between sub-models via other interface variables.

An adage in system-of-systems engineering is that “the design of the system is the design of the interfaces.” This is also true in the design of hybrid simulation models: choosing the interfaces and boundaries plays a major role in the model’s ability to provide clear, useful insight, and is done in an iterative fashion. The goal in interface identification is not to identify every possible interface among sub-models, but rather to identify the key interfaces among views in order to isolate interactions with the greatest effect. Iteration with later steps may be required to identify interactions that are missing as well as those that are superfluous. The goal is to develop an architecture that mirrors the structure of a nearly-decomposed system described by Simon (1958) (see Chapter 2), rather than a tightly interwoven web that is difficult to test and evaluate.

There are different classes of variables and events used in various simulation methodologies, and not all of them lend themselves to use as interface variables. Table 4-2 highlights some of the classes of variables that lend themselves to use as either input or output interface variables for three common simulation methods. While most classes of variables can be used as inputs, not all can be used for outputs. For example, a rule from an agent’s schema cannot be used as an output from an agent based model. It is the measure of the action of the agents that would be an output, rather than the rules that generated the behavior.

Table 4-2: Classes of variables that can be used as interface variables for three different simulation methodologies

	Inputs	Outputs
<i>System Dynamics</i>	<ul style="list-style-type: none"> • Stock size • Flow rates • Causal Variables • Events that shift behavior regime 	<ul style="list-style-type: none"> • Stock size • Flow rates • Causal Variables
<i>Agent-based Models</i>	<ul style="list-style-type: none"> • Agents' schema can be written to use any variable, event, trigger 	<ul style="list-style-type: none"> • Statistics on emergent agent behaviors
<i>Discrete event simulations</i>	<ul style="list-style-type: none"> • Process variables (size, time, frequency) • Events 	<ul style="list-style-type: none"> • Process variables • Process statistics on performing process • Events

Information gathered about the endogenous, exogenous, excluded, and interface variables is compiled into a Model Boundary chart for each sub-model for future reference and use during model construction. Table 4-3 provides an example of a model boundary chart for the organizational view sub-model from the TechSys case study that will be developed in the second half of this thesis. This sub-model, from inspection, takes in a number of exogenous inputs to ultimately calculate a single metric, inter-divisional understanding, which is an assessment of how well different operating units within the enterprise understand each other. Such a chart is very useful to quickly convey the boundaries, interfaces, and some structure to the modeler and other stakeholders before modeling begins, and can serve as a communication tool after modeling is completed.

Table 4-3: An example Model Boundary Chart

Endogenous	Interface (outputs)	Exogenous (Inputs)	Excluded
-Physical connectivity	-Inter-divisional	-Corporate IT	-Geographical
-Logical connectivity	understanding Metric	expenditures	distribution of OUs
-Number of shared/similar customers		-Strength of Corporate communications	-Leadership quality National monetary policy
-Shared processes		-Personnel Mobility between OUs (hr/mo)	
-Shared infrastructure		-Financial teaming	
-Number of inter-OU personal relationships		incentives	

4.3.7 Step 7: Create a sub-system and hybrid-level diagrams

In addition to selecting simulation methodologies to model each architectural view, the modeler must also choose how to connect all of the sub-models together to form the hybrid model. While a hybrid model can be conceptualized as a set of independent interacting sub-models, as shown in Figure 4-1, it may be easier to think of the model from the perspective of one of the sub-models as a controlling, dominant view. This is especially true if the enterprise behavior that the hybrid model is trying to capture and analyze are centered on a particular view.

For example, a hybrid simulation model might be intended to answer a process question that requires feedback and input from an organizational view and a knowledge view. Instead of implementing three completely separate sub-models of process, organization, and knowledge, the model can be implemented as a “process model” that has organizational and knowledge sub-models embedded in it that inform the operation of the process. Likewise, a hybrid model with behavior centered on a complex organizational agent-based model might be

implemented as an agent-based model with other sub-models serving as decision rules for the individual agents. The agent's resulting behavior would be back into the schema's sub-models to form feedback. See Figure 4-2 and Figure 4-3 for illustrations of this concept.

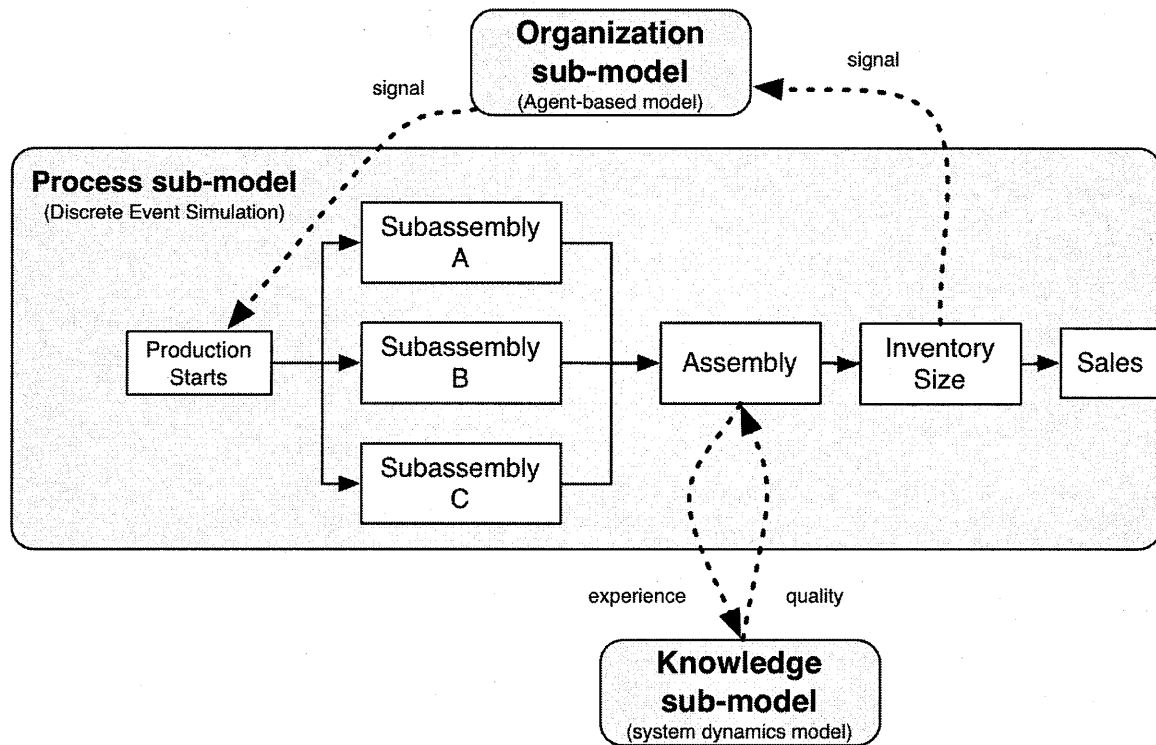


Figure 4-2: A hybrid model centered on a discrete event process sub-model (square boxes with solid lines) with connections to knowledge and organization sub-models (rounded boxes with dotted lines).

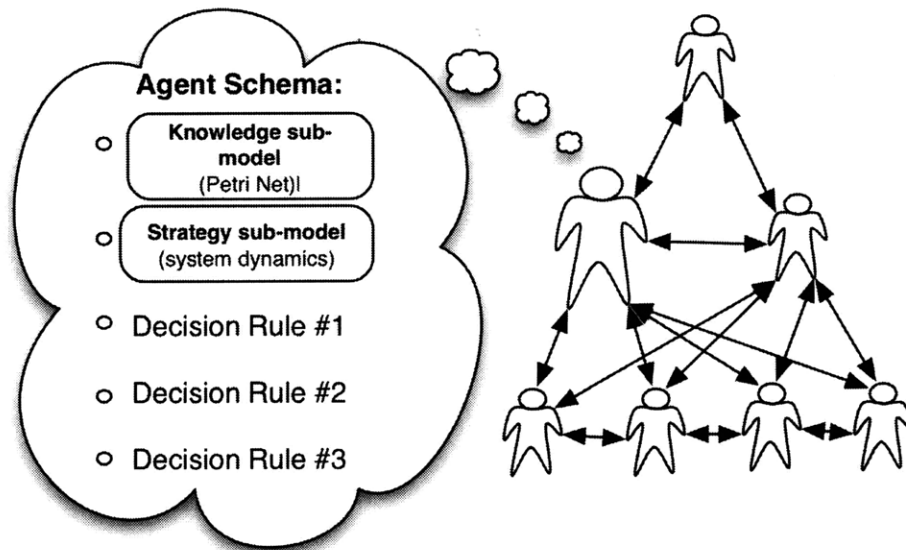


Figure 4-3: A hybrid model centered on an agent based model, where each agent's schema contains a knowledge sub-model and a strategy sub-model in addition to its other decision rules.

Most modelers will find that the hybrid simulation model will be more easily understood and utilized by its users if one of the sub-models is chosen to serve in a coordinating role, with other views informing that sub-model. The focus on a particular sub-model to frame the execution of the hybrid model is a matter of perspective, as the model can be translated such that any one of the other sub-models can serve in this role (or none at all). This translation is performed so that the dynamics that underlie the question the model is trying to answer can be easily highlighted and more easily understood upon quick inspection. In choosing a sub-model to serve in a coordinating role, the modeler must revisit the question the hybrid model is trying to answer to determine the focus of the simulation. For example, is the hybrid model fundamentally trying to answer a process question, or an organizational question? A question regarding the establishment of management incentives would do well to focus the hybrid model on the organization view sub-model, while displaying input from the other sub-models, rather than focusing on a knowledge sub-model, for example.

The high level interaction of sub-models can be depicted using a sub-system diagram in conjunction with a top-level diagram. The sub-system diagram is a high level graphical depiction of how each sub-model is structured and how all of the sub-models are coupled to produce the hybrid simulation. This step is very useful not only for planning the creation of the model, but also for communicating the model to stakeholders both for evaluation and use. A sub-system diagram clearly shows the boundaries and relationships among components in the enterprise architecture, and in essence serves as a static model that forms the basis for the creation of a simulation.

The top-level diagram details how the sub-models are connected together at a macro-level. This top-level diagram should be a reflection of the enterprise architecture itself, as it relates to the problem being modeled. This diagram is useful in “telling the story” of a model, graphically demonstrating the interactions between different views and components of the architecture.

The sub-model level diagrams are graphical depictions that represent the observed or hypothesized structure of the dynamics for each view. The nature of these depictions will vary greatly depending on what simulation method is employed by each sub-model. Some simulation methods, such as system dynamics, have an established system for creating such diagrams. In this case, causal loop diagrams are used to describe feedback structures, and stock and flow maps are used to illustrate the structure of the models (Sterman, 2000). Process simulation models often have formally defined languages and systems, such as Integrated Definition for Functional Modeling (IDEF0) that can be employed to describe the various interconnections. (Although, for purposes of the sub-model diagram, a less formal flow chart describing the process should suffice). Agent-based models do not have standardized static depictions. In these cases, the modeler must employ a bit of creativity in depicting the structures being modeled: for example, by depicting the interaction of agents and their rule sets to produce behavior. The critical component for these diagrams is

ensuring that the inputs and outputs are identified at the boundaries, and the nature of the interactions between the endogenous, input and output variables is shown.

4.3.8 Step 8: Implement the Hybrid Simulation Model

With all of the work done in the previous steps, what remains are the following tasks: (1) quantify the variables, (2) numerically estimate the relationships among variables, (3) create decision rules, (4) identify the initial conditions of the system and, finally, (5) implement the simulations. The necessary data for completing these tasks should be obtained while working with the stakeholders and subject matter experts of the enterprise. Wherever possible, quantifiable data should be used, ideally the same data (enterprise metrics) that are used to assess and manage the enterprise (e.g., quality indicators, inventory turns, ROIC, etc.). Often, the data needed for the model will not be readily available from the enterprise and must be determined via other means, such as statistical estimation or through interpolation from other sources. When quantitative data cannot be obtained through previous measurement or from system analysis, it may be estimated from expert assessments. Whenever variables are estimated, they should be later tested to determine how sensitive the model is to their value. When the model exhibits high sensitivity, greater pains should be taken to ensure that the data is accurate.

In cases where quantitative data are not available, qualitative measures should be used in concert with stakeholder input. For example, stakeholder surveys can be used to determine qualitative relationships using Likert scales, with clearly defined characteristics for each value assigned (e.g., 1 = no interaction whatsoever, 2 = infrequent interaction, 3 = occasional interaction, 4 = frequent interaction, 5 = interdependent interactions). The survey data should be incorporated into the rest of the model by defining relationships for each value level based on historical observations (i.e., for available data, frequent interaction

provides a 10% productivity boost over no interaction at all). This can be a resource intensive undertaking, so such efforts should be determined to be truly necessary before expending the energy to obtain the data.

After the variables and relationships have been quantified, the sub-system models should be developed. Preliminary model tests such as boundary adequacy tests, parameter assessment, and dimensional consistency should be conducted on the sub-models to ensure that they are operating as intended before integrating them into the hybrid model. Time spent testing the sub-models is extremely worthwhile; once the sub-models have been integrated together, it becomes much harder to track problems across model boundaries.

Once the sub-models are working as intended, they are then integrated using the top-level system diagram created previously as guidance. Integration of these different simulation methodologies can prove very challenging from a technical standpoint. For many years, the lack of a suitable, workable modeling environment for linking together hybrid simulation models has been a barrier to practical development of hybrid executable models in an enterprise context. Because different modeling methodologies are best suited to certain well-defined contexts (e.g., structure vs. rules, discrete vs. continuous, etc.), the software used to develop, estimate or execute models that employ different methodologies can vary substantially. Traditionally, hybrid models have relied on the use of "middleware" layers that attempt to take data from one sub-model that has completed execution and then input the result into another sub-model that is then executed.

This approach does not work as well when the sub-models must be run concurrently, especially when methodologies are vastly different in how they handle time (continuous, discrete, event). As an example, system dynamics is a continuous differential equation-based modeling methodology that requires a continuous-time modeling engine. Agent-based models and discrete event

simulations require a discrete timing engine for the simulation. Additionally, each of these methodologies has been developed over the years using different programming languages and software packages that usually have not been written and designed with consideration for interfacing with other methodologies.

Recently, however, advances in available modeling tools have made hybrid modeling more accessible to modelers with modest resources. More advanced middleware programs, such as ModelCenter,^{©28} have made interfacing disparate modeling packages and languages at least feasible. This approach is still difficult, however, and requires that software wrappers be written for each program or language that interfaces with the middleware, which can be a formidable obstacle.

Another recently developed modeling environment that shows great promise for hybrid modelers is AnyLogic.^{©29} AnyLogic has a timing engine that is written such that it can simultaneously execute continuous, discrete, and event-driven modeling engines inside a single environment. The environment is open and implemented in JAVA, allowing a modeler to insert arbitrary code and functions to create highly customized models. It also has a robust display capability. Connecting interfaces between disparate methodologies can be as simple as drawing a line between boxes (although more typically, interfacing is done in code, which remains straightforward). This software, possibly used in conjunction with another middleware package, has great potential for creating hybrid, multi-scale models that can model the emergent behavior found in enterprises.

²⁸ See <http://www.phoenix-integration.com/>

²⁹ See <http://www.xjtek.com>

4.3.9 Model Testing

Once the hybrid model has been created by assembling the sub-models, the long process of testing the model for utility and confidence begins. Model testing is used to build trust in the model to ensure that it is executing as intended and that its performance can be trusted for the purpose it will be used. These tests will evaluate the model's behaviors, assumptions, robustness, sensitivity, ability to replicate behavior in the modeled system, and performance. Following Sterman's guidance for models of complex systems such as these, the terms "validation" and "verification" are not used in model testing, as these imply a level of accuracy and certainty achieved with more conventional models that are impossible to have when modeling the architectures of complex enterprises that are constantly changing and adapting. Rarely are real-world enterprises stable enough that enough time-series data can be collected to truly verify a simulation. In the face of such challenges, other tests and approaches must be developed to build confidence in these models. This shift in terminology should not be seen as an opportunity to "let the model off the hook;" rather, it reflects the practical challenges of modeling complex socio-technical systems, and should remind the modeler of the extra effort that is required to build confidence in these models.

Fortunately, a number of tests can be run to increase users' confidence in the model for its intended purpose, and to increase understanding of the model's strengths and weaknesses. Sterman has assembled a large list of various tests that should be run on models of complex systems to aid understanding and build confidence, shown in Table 4-4. While not every test must be run for every sub-model (integration error is meaningless on an agent based model, for example), this table serves as a toolbox the modeler can use. Each test serves a different role in either building confidence or understanding the model more deeply.

It is essential that the model's ultimate users and stakeholders participate during the model testing process, because without their trust and understanding of the

model, it will not be used. Model weaknesses, such as hypersensitivity to variables, must be openly identified and shared to ensure that they can be addressed or later qualified when using the model. With transparency on the part of the modeler and stakeholder involvement, the model is much more likely to play a role in future decision making in the enterprise.

Table 4-4: Tests for assessment of dynamic models. Source: Adapted from (Sterman, 2000)

Test	Purpose	Tools/Procedures
Boundary Adequacy	Are the concepts tested by the model endogenous to the model? How does the model's output change as the boundaries are relaxed?	Model boundary charts, subsystem diagrams, inspection of model's equations;
Structure Assessment	Is the model structure consistent with the relevant descriptive knowledge of the enterprise architecture? Does the model conform to physical and logical laws?	Enterprise architecture, direct inspection of the model; Conduct partial model tests; Seek opinion of Subject Matter Experts;
Dimensional Consistency	Is each equation dimensionally consistent without the use of parameters having no real-world equivalent?	Inspection of model;
Parameter Assessment	Are the parameter values consistent with relevant knowledge of the enterprise architecture? Do parameters have real-world counterparts?	Partial model tests to calibrate models; Use judgmental methods based on interviews with stakeholders;
Extreme Conditions	Does each input parameter make sense, even when taken to extremes? Does the model respond plausibly to these shocks?	Model testing with extreme inputs;
Integration Error	Are the results sensitive to the choice of time step or numerical integration method?	Change time step;
Behavior Reproduction	Does the model reproduce the enterprise behaviors of interest?	Compute statistical measures of correspondence between the model and the actual enterprise;
Behavior Anomaly	Do anomalous behaviors result when assumptions of the model are changed or deleted?	Zero out key effects; Replace equilibrium assumptions with disequilibrium structures;
Family Member	Can the model generate the behavior observed in other similar enterprises?	Calibrate the model to the widest possible range of enterprises;

Surprise Behavior	Does the model generate previously unrecognized behavior? Does it anticipate the enterprise's response to novel conditions?	Keep accurate records of simulation runs over a wide variety of parameters; Resolve discrepancies between model behavior and the understanding of the enterprise's behavior;
Sensitivity Analysis	Do the numerical values, behaviors, or implications of the model change when parameters, boundaries, and levels of aggregation are varied over the range of uncertainty?	Perform univariate and multi-variate sensitivity analyses; Use optimization experiments to explore model sensitivity to a wide range of inputs;

Model testing should be a highly iterative step in the modeling process. It is likely that changes to the model structure will be necessary as the result of analyzing a boundary adequacy test or extreme condition test. In some cases, it may be necessary to rework a substantial part of the model as a result of a test that uncovers a deep flaw. In other cases, it may simply be necessary to point out limitations of some conclusions drawn from the model due to model biases or uncertainty.

4.3.10 Step 10: Policy Design and Evaluation

Policy design and evaluation is the act of determining a course of action (a policy) that can be implemented to achieve a desired behavior in the enterprise. After building confidence in its usefulness, the model can be actively used to investigate various management policies by varying the model's inputs and modifying aspects of the architecture in an attempt to improve model performance. In the context of a simulation model of enterprise architecture, this step is to be used to evaluate the current architecture, and to conceive and develop new architectures that can address weaknesses in the current enterprise architecture.

Several of the modeling tests from the previous step can inform the creation of new enterprise architectures. For instance, sensitivity analysis, as well as

parameter optimization, can be used to explore the range of enterprise behaviors given the enterprise architecture. In cases where the parameter space is simply too large to exhaustively explore and map, heuristic optimization techniques such as genetic algorithms, simulated annealing, or other heuristic methods can be used to identify the highest performing combinations of parameters that may maximize an objective function for the enterprise such as profitability. The results of any such optimization should then be further investigated to test for robustness to changes in the environment or sensitivity to inputs.

In addition to performing tests to investigate the impacts of changing input parameters, other tests, such as scenario and “what if” analyses can be used to determine the effects of changing environments or enterprise architecture itself. New processes can be designed, delays altered and incentives modified to evaluate the effectiveness of various ways of addressing problematic behaviors.

4.4 SUMMARY

The ability to simulate the breadth of an enterprise’s interconnected architecture gives enterprise managers and architects the ability to analyze its structure and behavior in ways that were not previously possible. The multiple views of an enterprise, each with their own context and dynamics, have proven to be challenges to those who have previously modeled the enterprise. The framework and process developed in this chapter are intended to help overcome these barriers and facilitate the creation of simulation models of enterprise architecture that can provide the ability to address questions that cannot be addressed by other simulation models such as “What are the implications to my process performance and strategic performance if I change to a new organizational structure?” or “how will my organizational performance be impacted by a change in government regulations?”

Such an approach to modeling enterprise dynamics has the potential for impact that can lead to greater understanding of the enterprise, its architecture, and its performance. In order to realize this potential, the approach needs to be tested and evaluated in a real world setting to build confidence in its capabilities and performance. The following three chapters will take this process and apply it to create a hybrid simulation model of the enterprise architecture of a large, real-world enterprise faced with challenges as they align their enterprise architecture with their strategy to meet future challenges, and demonstrate the unique value of this simulation approach for as a tool for senior management.

Chapter 5: THE TECHSYS CASE STUDY

The value of a framework for creating hybrid simulation models of enterprise architecture comes from its practical application to create a simulation model that is useful to an enterprise struggling to understand its own enterprise architecture. The goal of the next three chapters is to demonstrate the value of the framework for creating enterprise architecture simulation models developed in Chapter 4 by applying it to the real-world architectural concerns of a multi-billion dollar aerospace/defense enterprise. This practical, proof-of-concept application of the framework to create a simulation model should: (1) demonstrate that the framework for creating hybrid simulations of enterprise architectures is useful in guiding the creation of a hybrid enterprise architecture simulation model and overcoming barriers to implementation, and (2) demonstrate that such simulation models can be used to answer questions and provide insights into the architecture that traditional techniques would not be able to adequately address.

This chapter will focus on introducing “TechSys” as a case study and present the history and recent development of its enterprise architecture in support of a new enterprise strategy for growth. Chapter 6 will describe how the framework was applied to develop a hybrid enterprise architecture simulation that helped TechSys management understand to what extent their enterprise architecture supported their strategic goals. Chapter 7 will describe how the model was run, tested and analyzed. It will show how the model was used to test TechSys's own hypotheses and eventually create a new, alternative architecture that would be more likely to provide the goals TechSys has established for itself.

5.1 THE TECHSYS ENTERPRISE

“TechSys”³⁰ is a multi-billion dollar enterprise in the aerospace and defense sector that has undergone a significant transformation of its enterprise architecture in the past several years. TechSys, along with the entire aerospace/defense sector, had gone through several rounds of acquisitions and mergers in the wake of the Cold War and saw declining federal defense budgets on the horizon. The industry has since largely consolidated into two clusters: one cluster of large systems integrators, and another cluster of suppliers to those integrators. In terms of size, TechSys was near the middle of the industry; while it was not a small supplier, it was also not one of the large system integrators. It faced a difficult strategic decision: how would it strategically grow its business in such an industry? Did it have the right organization, processes, knowledge, and supporting infrastructure to pursue a new growth strategy?

TechSys is what has been termed an “enterprise of enterprises” (Sgouridis, 2007); it is an enterprise comprised of several constituent enterprises (operating units) each working together in a connected fashion towards a common goal.³¹ This was not always the case, however. At the time of its formation in the 1980s, TechSys was structured as a classic holding company comprised of completely autonomous operating units whose only common bond was the virtue that they were each in the aerospace/defense sector and that they shared a common owner. In this early phase of its existence, using the term “enterprise” to describe TechSys would be a stretch of the term, as there was no common

³⁰ “TechSys” is a pseudonym used to protect the identity of the enterprise in the case study. Much of the analysis in the model is sensitive, so measures were taken to protect TechSys’s identity and disguise any identifying data.

³¹ Within the case study, use of the term “enterprise” will refer to the TechSys division, rather than to one of its operating units

purpose, market or strategy uniting its operating units. TechSys's operating units (OUs) were not aligned in any strategic sense nor did they communicate with each other to any great extent apart from reporting their financial position.

TechSys itself is a wholly owned subsidiary of a larger conglomerate known here as "BigTechs." BigTechs had grown aggressively through mergers and acquisitions across a variety of sectors, ranging from high-tech aerospace/defense to low-tech durable consumer goods. TechSys was formed out of BigTechs' aerospace/defense holding, growing and contracting as new operating units were acquired or divested by BigTechs. For many years, the relationship between BigTechs, TechSys, and the operating units of TechSys was purely financial; BigTechs, the corporation, would establish financial goals for TechSys, the division, which TechSys would pass on to its operating units in a fairly typical holding company fashion.

Corporate

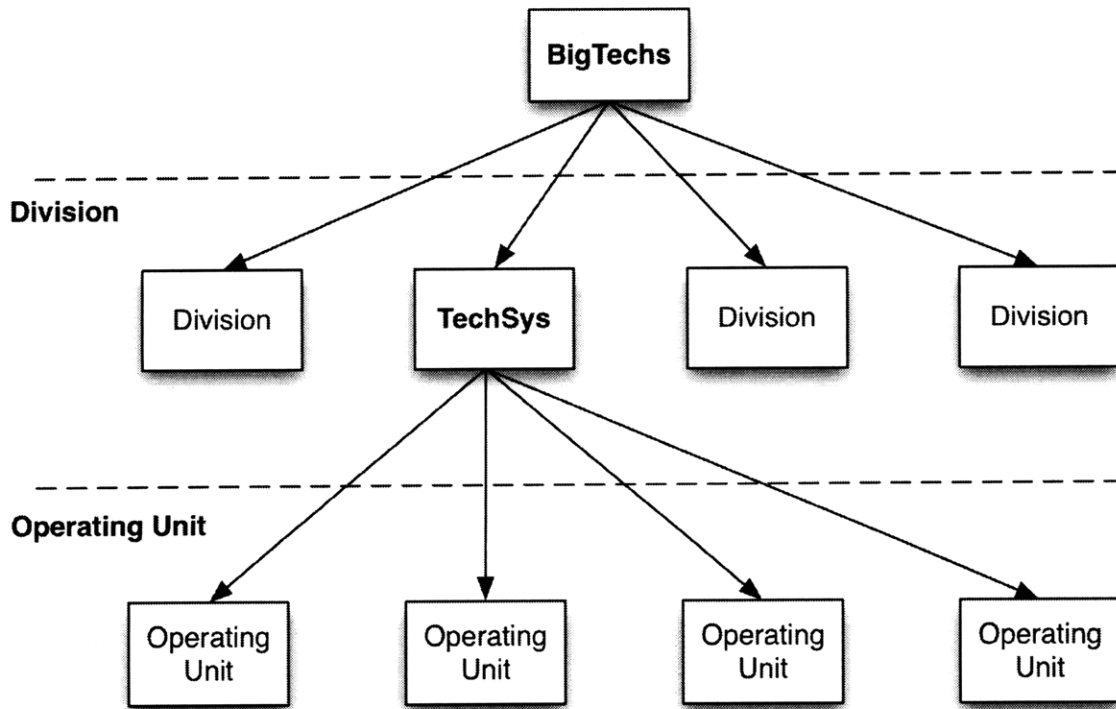


Figure 5-1: The organizational structure of BigTechs and TechSys

In 2006, TechSys sowed the seeds for a new strategy of future growth based upon increasing its competitive position in the marketplace by encouraging cooperation between its operating units to produce integrated product offerings. Looking at the businesses of its operating units, TechSys realized there were several opportunities for development of new integrated products, and hoped to create an environment that allowed to TechSys began the process of transitioning to a new enterprise architecture that would facilitate the new strategy. After two years of incremental development and implementation, the leadership at TechSys wanted to know if their current enterprise architecture was capable of achieving their growth goals and how it could be better managed to maximize growth. They were willing to participate as a case study participant in order to see if a hybrid enterprise architecture simulation model could help them answer some of their questions about their enterprise's performance and possibly help them think about their enterprise in new ways.

5.2 CASE SELECTION

TechSys proved to be an appealing candidate for application of the hybrid enterprise architecture simulation framework for several reasons. The primary reason was the nature of its problems: TechSys recently changed its enterprise architecture in order to effect a behavioral change in the enterprise, and those changes have *impacts across many views of the architecture*. TechSys's challenges contained aspects pertaining to the strategy, organization, process, knowledge, policy, and IT views of the enterprise architecture in a highly interconnected fashion and required a holistic approach to analysis that treated the enterprise from a system perspective. Some of the questions that TechSys had potentially could not be addressed using traditional modeling and simulation approaches without reducing the scope of the model to the point that key enterprise dynamics could be missed. An enterprise architecture-based

approach provided the ability to analyze the enterprise's behavior from multiple perspectives and a simulation model capable of helping visualize its performance and management tradeoffs in a way that had before not been possible.

TechSys also was a good candidate for the hybrid enterprise architecture simulation case study because of its position in its enterprise transformation efforts. It had reached a point where many of its processes and its strategies were becoming modelable, in the sense that they were documented, repeatable, and in use across much of the enterprise. This had not previously been the case, as earlier processes and structures had been ad hoc. Without a stable and repeatable architecture, the architecture could not be modeled.

Although TechSys had not formally documented its enterprise architecture or previously used an enterprise architecture framework to guide its architecture's development, it had addressed its own design using terms, concepts and practices that are compatible with enterprise architecting by focusing on the alignment of strategy, organization, processes, and other such enterprise dimensions³². Given this background, along with in-depth access to its data and active support from multiple levels in the enterprise, TechSys provided a rich environment to test both the theory and practice creating hybrid simulation models of enterprise architecture.

5.3 HISTORY OF TECHSYS

The 1990s were extremely difficult for TechSys; in the wake of the Cold War, the defense industry went through wave after wave of consolidations as the defense budget was dramatically reduced. TechSys transformed its organizational

³² There was no formal conceptualization of enterprise design associated with a External/Policy view, Knowledge view, or Services view, but this was elicited over the course of interviews with subject matter experts.

architecture from a typical rigid hierarchy of the time to a very flat, team-based structure in an effort to be more flexible and nimble during an era of scarce resources. While some TechSys veterans stated that this structure may have saved TechSys at the time, when the industry began to grow once more at the turn of the century the team-based structure was seen to be a barrier to future planning, management and growth at TechSys.

TechSys brought in a new CEO in the mid-1990s with a strong strategy and process background. He sought to increase the competitiveness of TechSys by focusing on process excellence and looking for ways to share best practices and common functions across the operating units. TechSys's strategy at the time, similar to many of its competitors in the sector, was to grow through increased efficiency, both in structure and operations. There were heavy investments made in creating common financial, strategic and human resource processes across the division. Many of these changes were being driven from the top down from BigTechs, while others were grown internally at TechSys. TechSys deployed a Total Quality Management initiative in 2000 to improve its own processes, and later lead active Lean and Six Sigma programs across the enterprise that continue to play a strong role in the enterprise today.

5.3.1 A New Strategy Requires a New Enterprise Architecture

In the mid 2000s, TechSys began to take a more aggressive approach to growth. It found itself occupying a "middle ground" in the defense sector: it was neither a large system integrator nor a small supplier of parts. TechSys was too large and possessed too many technical capabilities to be considered a simple supplier, but it did not yet have the structure or knowledge that would allow it to perform in the higher-profit role of system integrator/supplier. Its four operating units had combined annual revenue on the order of one billion dollars, but there was little complementarity between them; they were essentially four separate enterprises

with shared overhead. This limited TechSys's potential for strategic growth as a division and kept it in the role of a high-tech supplier in multiple markets rather than in the role of system provider.

TechSys's CEO had a vision to create a true enterprise from the division with a common purpose, goals, and complementary strategies amongst the operating units. He desired to grow TechSys through strategic acquisitions, adding companies that complemented the business of existing operating units. Once a portfolio of complementary companies had been developed, the new strategy called for the operating units to work together on new products that took advantage of the collective knowledge and capabilities across TechSys, allowing the enterprise to move away from the role of supplier in multiple markets and into the role of integrator that could deliver systems of its own, growing into new profitable markets that were previously unavailable.

Before beginning its acquisitions, however, TechSys knew that it would need a new enterprise architecture that would facilitate not only the addition of several more operating units, but that would also enable better communication and collaboration between them. In 2005, TechSys worked with a business consultant to develop a new organizational architecture that would be better aligned with customer needs and could better accommodate growth through acquisition. While TechSys did not use the term "enterprise architecture" to describe these efforts, its planning very closely resembled the practice of enterprise architecting: they developed a new strategy for the enterprise and then sought to put into place organizations and processes that supported that strategy.

Organizational Structure

The new organizational architecture had two very significant changes. First, it created a very strong general manager position at each operating unit with profit and loss responsibility and decision-making authority that had been previously

distributed amongst the project team leaders. Previous to the general manager position, there was no single face of the operating units to the customer. The general manager provided this single face. Additionally, all general managers were given the title of Senior Vice President, and were included on the Executive Leadership Team, taking part in TechSys-wide decision making. With this role, the General Managers were given financial incentives aligned with the performance of the corporation and the division over the performance of their own operating unit.

The second major change was a new matrix-like structure that staffed many senior positions at the operating units with “dual-reports”—individuals that reported both to the general manager as well as to a functional executive at TechSys. The dual-reports were intended to serve as two-way conduits for ideas and best practices across the enterprise, and to be the catalysts for greater collaboration between the operating units. The dual reports would know their peers at other operating units, providing for horizontal as well as vertical communication pathways across the enterprise. Figure 5-2 is an illustration of the organizational structure at TechSys after the reorganization of 2005-2006.

In Figure 5-2, the operating units are shown in the four vertical boxes at the bottom of the figure. Each operating unit has a dual report position in each functional area (shown with a small diamond) that is accountable to both the General Manager of the operating unit as well as to the corresponding functional executive (senior vice president) at the TechSys division level, shown with single headed arrows. The thin horizontal boxes across the operating units are intended to show that there is some commonality across the operating units within the functional areas.

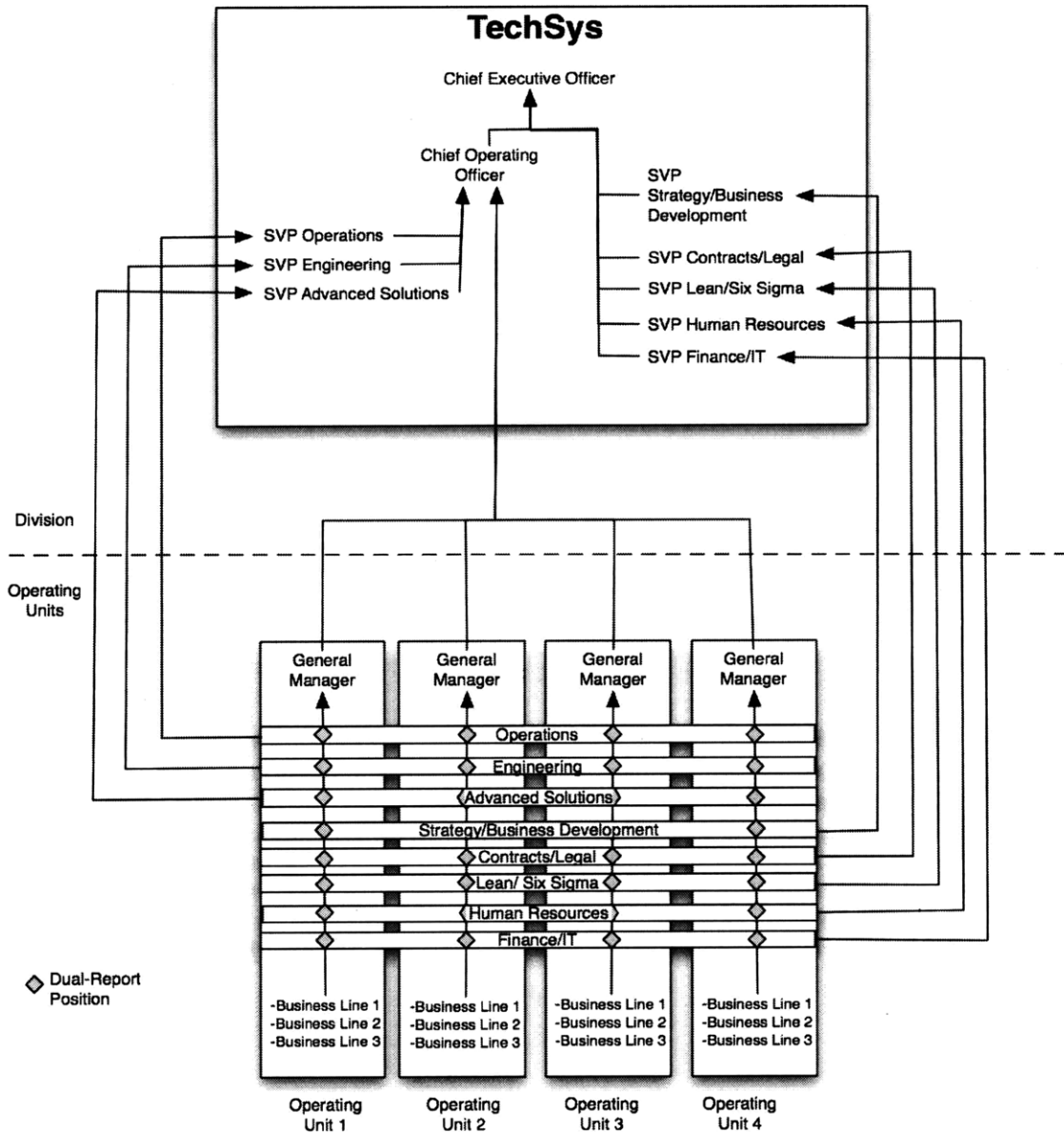


Figure 5-2: The organizational structure of TechSys after 2005

The general manager has ultimate budget authority for all allocations at his operating unit, while the functional executive at the TechSys would roll out enterprise-wide initiatives and advocate common processes and best practices. There is a fundamental tension in this architecture between the general managers with profit and loss responsibility for their operating unit and the function executives pushing strategically mandated programs and tasks that are dependant of securing funding from each general manager. Between these

conflicting roles lie the dual-reports, who are accountable to both parties and are intended to resolve conflict and help make tradeoff decisions, such as for allocation of resources among functions, operating units, and TechSys. As to be expected, this architecture presented its own challenges, and there were minor changes introduced over the subsequent years to reduce some of the tension in this structure while keeping the dual-reports in a position to facilitate communication and help analyze tradeoffs between the operating units and the division.

Incentives

In addition to a redesign of the organizational structure, the incentives of many of the high level managers were updated in the rearchitecting process. The incentive structure was modified for all senior positions, from the directorate level and higher, to ensure that the majority of their performance compensation package would be determined based on the performance of BigTechs and TechSys, with little emphasis on the performance of the local operating unit. This was intended to foster a culture where decisions were made based upon a "what's best for the enterprise as a whole" philosophy. The performance bonuses of the mid-level managers at operating units continued to be determined by the general manager, and the pool of available money for performance bonuses was based upon the financial performance of the operating unit.

Advanced Solutions

One unusual aspect of the TechSys organizational structure is the integrative role played by the "Advanced Solutions" group in each operating unit. At first glance, Advanced Solutions serves as the research and development function at TechSys, but it plays a larger role than most research and development functions typically do. There is a local Advanced Solutions group within each of TechSys's operating units, as well as a sizable group that works solely at the TechSys division level on behalf of all of the operating units. In addition to conducting technical research and development across the enterprise, Advanced Solutions is tasked with helping the enterprise strategically position itself for the future, in

terms of products, technologies, and processes. Advanced Solutions plays some role in developing strategy, identifying new markets, establishing business cases, identifying future customers, and developing technical and product roadmaps that establish the future direction of the enterprise. This overlaps with the traditional functional roles for strategy and business development; typically, Advanced Solutions focuses its efforts on unproven future programs and technologies, leaving the traditional functional roles to focus on proven markets and customers. This allows Advanced Solutions to function as an incubator for new ideas, markets, strategies, and technologies across the enterprise.

The Advanced Systems group at the TechSys division level is focused on technologies, tools and strategies that will facilitate closer collaboration between operating units or benefit the division as a whole, as opposed to a single operating unit. For example, when new operating units were acquired, the integration effort was led by Advanced Systems. In a way, Advanced Systems at the division level functions as its own small operating unit that performs all of the front end work that leads to new business that the other operating units will eventually pursue and capture.

5.3.2 Developing Common Processes and Metrics

In addition to the rearchitecting work done on the organization, the processes at TechSys also received attention as part of the realignment with strategy. Before 2005, the operating units at TechSys had almost complete independence with regards to their local processes, as each operating unit operated in a different market with different products and technologies. Operating units were encouraged by the Senior Vice Presidents of Operations and Engineering/Program Management to use best practices while developing their processes, but they had autonomy in doing so.

With the creation of the new strategy for cooperation between the operating units, a new effort was initiated to develop some process commonality where applicable across operating units. The challenge was to create common processes where useful to TechSys, while retaining unique processes where commonality was unnecessary for cooperation among operating units. This proved to be a formidable task, as there was tremendous resistance from the operating units to any changes to their existing processes. Efforts were made to focus standardization efforts on enabling processes such as program management, general lifecycle management and strategy development with some success. For core mission-oriented processes, emphasis was placed on creating *compatible* processes, rather than *common* processes, allowing operating units to smoothly interact when needed. Rolling out new processes and tools throughout TechSys has become an ongoing activity, rather than a sudden change.

In addition to efforts to increase process commonality and compatibility, a significant amount of effort was spent developing a new performance measurement system that supported the new strategy and assessed TechSys's performance against its strategy. The new metrics system had to balance both individual financial goals of the operating units against other goals of the enterprise, such as Six Sigma certifications or jointly pursued business opportunities. After new metrics were identified, a thorough process known as the Goal Deployment Process was established for reporting these back to the Executive Leadership Team, which monitored the enterprise's progress towards its strategic goals.

5.3.3 TechSys Makes its Acquisitions

With a new organizational architecture in place and processes under continuous improvement, TechSys made its first acquisition in 2006, acquiring a company

whose technologies strongly complemented the work of TechSys's other operating units. A TechSys integration team was established to identify potential projects and programs that the new operating unit could collaboratively develop with the existing operating units.

As to be expected, there were growing pains as changes were made, often due to mismatches in the architectures and cultures between TechSys and its acquired operating unit (the new acquisition had a different business model and relationship with its customers, and its markets demanded processes that were very different from those already in place across TechSys). The integration team hoped for a two-way transfer of best practices, and was eager to learn about the new operating unit's best-of-class processes and roll them out at other operating units.

Only one year after this acquisition, TechSys then made a much larger acquisition in late 2007 that doubled the annual revenue of TechSys. Many potential synergies existed between this new operating unit and the existing operating units. A similar transition team was established based out of Advanced Systems. The cultures and processes between TechSys and this acquisition were more closely aligned, reducing the effort required to integrate them into the TechSys enterprise.

With this addition, it was now clear to the market that TechSys had the resources and technologies at its disposal to deliver some integrated systems products and possibly enter new markets. There remained a serious challenge, however: would the new TechSys enterprise architecture actually be able to support a collaborative environment between its operating units? Could the operating units, which for years had been kept operationally independent, work together to create new value?

Table 5-1 provides an overview of the changes that were made to TechSys' enterprise architecture to increase in ability to meet the strategic goals that TechSys had established for growth.

Table 5-1: List of modifications to the TechSys Enterprise Architecture to align with new strategy

-
- Created General Manager position at each Operating Unit (OU) with profit/loss responsibility
 - Created a dual-reporting organizational structure between OUs and functional executives
 - Developed common overhead and management processes and metrics across OUs
 - Created an Advanced Systems function that cut across OUs
 - Acquired new OUs with complementary technological capabilities
-

5.3.4 TechSys's Challenges

When TechSys first began participation in this case study in 2006, it was actively analyzing its architecture and its ability to effectively facilitate collaboration across operating units. The new organizational structure was established, many new processes had been deployed, and the new acquisitions were about to be made. What was not certain, however, was if the new architecture would actually support its intended goals of collaboration.

Historically, there had been very little collaboration between operating units. There was only a single example of a collaborative project before this transition. Unlike most non-defense enterprises, TechSys is hindered from completely open collaboration and resource-sharing among its operating units by the Federal Acquisition Regulations (FAR). The FAR established strict accountability guidelines that make it difficult to transfer personnel and resources from one

operating unit to another on a temporary basis. These rules eliminate most financial incentives for collaboration among operating units over collaboration with external organizations. The net result was that the TechSys operating units had little experience working together on joint endeavors. Other than a handful of functional executives and some dual-reporting positions, there were very few professional relationships that spanned operating units. The exception to this was that operating units could more easily “borrow” people from the TechSys division level, but this pool of people was fairly small.

The senior leadership of TechSys had established daunting “stretch” growth goals for the enterprise: doubling its size every four years, a growth rate of almost 19%. Assuming even optimistic growth forecasts, it would be hard to imagine the operating units generating enough business in their existing markets to meet these growth goals, especially in the face of declining demand as the wars in Iraq and Afghanistan begin to drawn down. TechSys intended to get some of this growth through what it termed “synergy”: collaboration between operating units in ways that strengthened the collective position of the enterprise and allowed it to enter into new markets. The question remained, however: would the new enterprise architecture facilitate collaboration of operating units in the pursuit of new business? Would the operating units have the incentives to collaborate?

In the wake of the acquisitions, a “synergy team” within Advanced Systems at the TechSys division level had been tasked with identifying the barriers to cooperation between the operating units. They worked systematically to identify individual barriers to synergistic collaboration in many different areas, ranging from culture to financial processes, but had no way of understanding the magnitude of the impact of each barrier on the performance of the system, or how these barriers may be interconnected.

The “synergy team” needed the ability to analyze the enterprise architecture holistically and quantitatively in order to understand the impact of each barrier on the enterprise’s performance, and what changes could be recommended to improve synergy among operating units. Due to the close alignment of interests, the leader of the “synergy team” (the Director of Strategy in Advanced Systems) took a direct interest in the creation of the hybrid enterprise architecture model for TechSys. He saw that the simulation model had great potential to help TechSys better understand its own architecture and the potential capabilities of that architecture. As a result of this fortunate alignment of timing and purpose, the Director of Strategy played a critical role in the development, testing, and application of the new simulation model from the time of the initial stakeholder interviews through model testing and analysis.

TechSys proved to be a good test case for the first application of the hybrid enterprise architecture framework. The combination of the enterprise-wide nature of its problems tied to its architecture, coupled with the timing of the case was a near-ideal opportunity to demonstrate the ability of hybrid enterprise architecture simulations to answer tough enterprise problems that span multiple views in an imperfect real-world setting. There was support from multiple levels of management, along with ready access to both data and people. From the CEO down, TechSys was interested in what insights simulating their architecture might provide to them.

After TechSys was chosen as a case study and stakeholder buy in was obtained, work began to implement the process developed in Chapter 4 for hybrid simulation modeling of enterprise architecture to build a functional simulation model for TechSys. The next chapter will describe the implementation of the model using this process, highlighting the obstacles, solutions, and lessons learned from the process.

Chapter 6: THE APPLICATION OF THE FRAMEWORK FOR CREATING HYBRID ENTERPRISE ARCHITECTURE SIMULATIONS

TechSys was in good position to benefit from the creation of a hybrid enterprise architecture simulation model. It had recently undergone a series of changes to its enterprise architecture, and was trying to better understand the connection between its enterprise architecture, its strategy, and enterprise performance. In order to create this model, however, much work was required. First, the problems that TechSys faced would need to be clearly articulated in a way that could be addressed using a simulation model. Additionally, a substantial amount of work laid ahead in documenting the enterprise architecture. Although TechSys had purposefully changed its strategy, organization and processes, it had not documented the changes in a single place or using a coherent framework. A large amount of research would be needed to document the enterprise architecture for use in the simulation model. The successful creation of the simulation model would depend on adhering to the framework and processes established in Chapter 4.

This chapter will describe how the framework from Chapter 4 was applied to TechSys in order to create a hybrid enterprise architecture simulation model that helped them address a specific set of architectural concerns. The structure of the chapter will mirror each step of the model creation process developed in Section 4.7 and shown in Table 4-2. It will begin at the initial step, *Documenting the Enterprise Architecture*, and continue through the eighth step, *Model Implementation* (the last two steps, *Model Testing* and *Policy Design and Evaluation*, are addressed in Chapter 7). Each section will describe how the framework was applied to create the TechSys model and illustrate the intermediate development products, such as model boundary charts and sub-system diagrams, culminating with the full development of the simulation model.

6.1 STEP 1: DOCUMENTING THE ENTERPRISE ARCHITECTURE

The first step of the modeling process developed in Chapter 4 is straightforward yet surprisingly difficult: document the enterprise architecture. In an ideal application of the modeling process, the enterprise architecture would have already been documented and could be used as the basis for building the hybrid simulation model. The documentation of the enterprise architecture would be done as part of the on-going functioning of the enterprise, and the enterprise architecture could be easily referenced when the need arises. Most enterprises today, however, do not maintain an up-to-date enterprise architecture, and if they do, it is highly focused on the implementation and structure of IT systems. This is also true for TechSys; although it had a coherent enterprise architecture, the architecture had not been documented using an enterprise architecture framework or captured using any disciplined means that united the many views of the enterprise.

Since the framework developed in Chapter 4 depends on the use of a documented, framework-based enterprise architecture, TechSys' *de facto* enterprise architecture needed to be documented using such a framework. For

this purpose, the Nightingale-Rhodes Enterprise Architecting Framework (NREAF), described in Chapter 2, was employed. (See This framework contains a set of views that span the enterprise, has a focus on interactions across these views, and is flexible enough to be adapted to a wide range of enterprise types. Its focus on strategy-driven design, centered on the organization, process and knowledge views well-suited the enterprise architecture independently developed and implemented at TechSys.

6.1.1 Initial Data Collection for EA Documentation

The initial steps towards documenting TechSys' enterprise architecture began with a series of open, exploratory interviews with the executive leadership team and the consultant that helped them design and implement the first stages of their new architecture. The CEO and COO of TechSys explained their strategic desire to move their enterprise away from a traditional, independent, holding company structure towards a more complementary and cooperative family of operating units. Such a family could leverage each other's capabilities to create a stronger competitive position for each operating unit. TechSys wished to grow through acquisitions that served to strengthen the competitiveness of the family, and desired an enterprise architecture that could accommodate the growth of the enterprise through acquisition as well as through complementary fit of the operating units. The early interviews with executive management focused the discussion on three key areas: (1) developing a strategy for growth of the enterprise; (2) developing an organizational structure that could support the growth, and (3) developing common, or at least compatible, processes across the operating units to support the strategy. These early, loosely structured interviews, along with data provided in the form of reports and presentations, served to give a high-level view of TechSys' overall architecture, and provided a launching point for further, more targeted investigation and development.

Initial interviews with the CEO and COO were followed by more structured interviews with the Executive Leadership Team, consisting of the Senior Vice

Presidents for Research and Development, Strategy/Business Development, Finance/IT, Engineering, Operations, Lean/Six Sigma, and Communication. Each interview lasted from one to two hours, and the interviewees provided supporting documentation in the form of presentations, reports, charts and process documentation where possible. This round of interviews was used to develop the basic content of the NREAF views and to provide direction for further investigation.

In the early stages of conducting the interviews with the Executive Leadership Team, a problem had not yet been articulated for the simulation model to address. As a result, these early interviews focused on capturing the larger enterprise architecture, without specific emphasis to any one area or issue.

After receiving some background information, each interviewee was asked to describe the view of the enterprise architecture that they were most closely connected with. Some positions, such as Operations, Strategy/Business Development, or Finance/IT fit neatly within boundaries of a single view. Other positions, such as Communications or Research and Development spanned several architectural views; in these cases, they described the views most pertinent to their duties. Each interviewee was asked to describe both the key structures and key behaviors associated with their associated views. The interviewees were then asked to identify the key interactions from their “primary” view with the other views in NREAF and were specifically asked about the key interactions identified for consideration by the NREAF.

In the case of the Strategy View, an example of a key interaction with the Knowledge View and the Process View would be the Technology and Product Roadmapping Process, where specific technical needs would be mapped to future products dictated by the enterprise’s strategy. This process produces documents that ideally would serve as the clear map between strategy and future knowledge requirements.

6.1.2 Enterprise Architecture Iteration

The TechSys Enterprise Architecture, as captured for use in guiding model development, should not be considered a complete documentation of the enterprise architecture. It is a high level architecture that falls short of a true architecture with sufficient detail for implementation of systems and processes. Because a more thorough documentation of the complete enterprise architecture would require the considerable effort of a team of individuals and a much broader array of stakeholders, this research effort captured the enterprise architecture broadly, and then more fully developed in areas that influenced the behaviors under study. While this allowed resources to be used efficiently, it also opened up the possibility that some areas of the architecture would not be documented in sufficient detail, and would require a second iteration to fill in any gaps.

Determining which areas of the enterprise architecture must be fully documented and which did not need to be fully documented was made difficult because some areas might have a significant but non-intuitive impact on the modeled behavior. If the net were cast too narrowly when initially collecting enterprise architecture data, the risk would be that an innocuous but critical structure or element of feedback could be missed. If the net was cast too broadly, then extra time would be spent collecting architecture data that might not be used by the simulation. In practice, there was iteration between architecture documentation and the modeling process; as gaps were uncovered during modeling, the enterprise architecture was revisited and further documented to capture structures, incentives, rules, and dynamics. This iteration would not be as essential if the modeling process began with an accurate, up to date enterprise architecture.

6.2 STEP 2: PROBLEM ARTICULATION

At the outset of the case study, TechSys did not have a single question that it wished to have answered using an enterprise architecture simulation model. Instead, there were a number of potentially valuable areas that could be explored

using the hybrid simulation modeling approach. Some of the issues and problems for TechSys that surfaced in the early stages of interviews included:

- Sources of friction related to the incentives and structures of the organizational architecture for personnel in dual-accountability (matrixed) roles;
- Barriers in the enterprise architecture to achieving collaboration between operating units; and
- Achieving transparent linkages from strategic planning to strategic execution, and monitoring execution that was in accord with the strategy.

After some thought with regard to what would prove to be both a useful topic for TechSys and a good candidate for a proof-of-concept case study, the topic area was narrowed down to understanding barriers in achieving collaboration between operating units. This was an active issue being addressed at TechSys, data was available, and it had the support of the Director of Enterprise Strategy and the Senior Vice President for Advanced Solutions. In a typical modeling scenario, the problem would be chosen based on need rather through closeness of fit with an assumed framework. Fortunately, in this case a topic was identified that was both of great importance and was a good candidate for a hybrid enterprise architecture based simulation model.

The initially identified topic, “barriers to collaboration,” was an expansive, “fuzzy” topic; it provided a general area for further research, but it was vague, lacked structure, and was not posed in a way that could be concretely addressed using a simulation model. The topic area was revisited with key stakeholders and was iteratively scoped into a more specific, defined problem following the guidelines established in Chapter 4 for problem articulation. These guidelines seek to establish and define:

- A clearly bounded root problem tied to the architecture;
- key input/output parameters;

- critical behaviors;
- time horizon, and
- appropriateness of hybrid simulation modeling

The next several sections describe how each of these key points was defined for the TechSys simulation model.

6.2.1 Identifying and Bounding the Root Problem

After the general topic for the simulation model had been chosen, the first step to problem articulation was to more concretely define and bound the topic into an addressable problem. The chosen topic, “collaboration between operating units” would need to be refined to determine *what* aspects of collaboration should be considered, and reviewed to determine that the area of interest was truly collaboration, and not something broader.

There are many ways collaboration can occur, such as sharing market information, sharing best practices, or exchanging personnel. In the case of TechSys, discussions uncovered that they were most concerned about how operating units could work collaboratively to pursue new business opportunities, and how this collaboration could be used to increase the competitive position of the greater enterprise. New business pursuit encompasses the activities ranging from the development of ideas for new business opportunities through everything required to submit and win a new business contract. During the most recent round of re-architecting, TechSys attempted to create an enterprise architecture that could facilitate collaboration between the operating units when pursuing new business opportunities, allowing them to win new business that they could not have won without collaboration.

TechSys uses the term “synergy” growth to describe the kinds of “business opportunities pursued jointly between two or more operating units that increase

the competitive position of the enterprise.” Synergy growth is growth that comes when two or more operating units work together to increase effectiveness in existing markets and pursue new markets. Synergy growth is contrasted by “organic” growth, which is growth that advances the existing business of a single operating unit without collaborating with any other operating unit. The combination of both synergy growth and organic growth leads to total new business growth for the enterprise. The challenge that the enterprise faces is twofold: (1) how does it enable synergy growth between operating units, and (2) what is the right balance between synergy and organic growth to maximize total new business growth across the enterprise, assuming synergy growth is possible? The enterprise has limited resources, and they must be allocated between pursuing synergy and organic growth.

Identifying the Root Problem

Given this, the original topic of “collaboration between operating units” has evolved substantially towards a deeper understanding of root problem of importance to TechSys. The goal of the enterprise is not simply to maximize synergy growth (increase collaboration); it is to maximize the *total* new business growth across the enterprise. This can be achieved through increasing synergy growth, organic growth, or a combination of the two. TechSys must make decisions regarding the allocation of resources to growth, but it currently has no tools or process that would allow it to determine the effectiveness of investments in promoting synergy growth and no way to understand the trade off between synergy growth and organic growth. One of the intended goals of the TechSys simulation model is to provide this capability.

To be genuinely useful to TechSys, the simulation model must focus on the total pursuit of new business opportunities, rather than a single component of growth. The enterprise architecture-based simulation model must help identify levers in the architecture to enable synergy growth as well as show the outcome of

strategies for resource allocation towards synergy and organic growth to achieve maximum total growth.

The dynamics of TechSys' architecture for new business opportunity pursuit and capture are complex, but the problem area can be well bounded for purposes of modeling. The dynamics of new business opportunity pursuit and capture are strongly driven by strategy at both a divisional and operating unit level, with local decisions made in an organizational context, following established processes, dependent on aligned knowledge requirements, supported by information technology, and bounded by external constraints. The simultaneous interaction of all of these factors across the enterprise architecture can make for complex behavior that cannot be analyzed without the aid of a model of the architecture. Fortunately, the problem area is neatly bounded by a handful of processes and a specific organizational structure with clear inputs and outputs. For all of these reasons, this problem was well-suited for evaluation using a hybrid, enterprise architecture-based simulation.

The key questions for TechSys surrounding the pursuit and capture of new business opportunities include:

- Can TechSys achieve its growth goals (both synergy and organic growth) given its current enterprise architecture with constrained resources dedicated to growth?
- How sensitive is the architecture to changes in resource allocation?
- What changes can be made to the architecture to improve growth opportunities given constrained resources?
- What combination of inputs should be used to best grow the enterprise?

6.2.2 Identifying Inputs and Outputs

After clearly stating the problem, the next step of problem articulation is to identify the key input parameters and the output of the simulation model. It is usually easier to determine the output of the model first, as there is often a clear idea of some quantity that must be maximized or minimized. This was true for TechSys; they sought to track the revenue and profits that arise from the capture of new contracts. Revenue and profits in this model can come from two sources: organic growth from a single operating unit, or synergy growth arising from contract awarded to two operating units cooperating on a contract.

Identifying the inputs for the model required an examination the controls that TechSys management uses to influence the enterprise's ability to pursue and capture new business, such as funding, headcount, and other resources. It also required a review of the process, to identify any other potential inputs that may not be in current use by TechSys. At an operational level, there were over a dozen possible inputs identified that had some impact at some point in the processes. Working with TechSys stakeholders, the large list of possible inputs were distilled down to five strategic resource inputs that management has direct control over and uses to influence the enterprise.

The first input was the percentage of all new business opportunities (ideas that may later become proposals) that contain synergy growth. This input can be thought of as a directive from TechSys to its operating units to have a minimum number of synergistic project proposals in a given year from its total pool of project proposals. This is the percentage of proposals submitted for consideration for funding and development that require collaboration with another operating unit in a new market. The assumption is that all other project proposals are organic projects, focusing on an operating unit's existing business areas.

The second, third and fourth model inputs are budget allocations. The first of these budgetary inputs is the size of Discretionary budget, a pool of money that TechSys can use to invest in future growth, allocated to them by the government. The Discretionary budget is divided between two subcategories: the bid and proposal budget and the internal research and development (IRAD) budget³³. The bid and proposal budget is used to pay for program managers and engineers to investigate and write proposals for new business, while the IRAD budget is used to research new technologies and processes for future programs. Although the government primarily determines the size of TechSys's Discretionary budget³⁴, it remains a key budgetary input parameter into the new business pursuit and capture process. The second budgetary input is something that TechSys directly has full control over: the allocation of the budget between bid and proposal activities and IRAD activities. The input is represented in the model as the percentage of the Discretionary budget is allocated to bid and proposal, with the understanding that the remainder is to be used for IRAD activities. This model input represents the emphasis that TechSys places on exploring opportunities in the future (IRAD), versus exploiting existing capabilities and opportunities (bid and proposal budget).

The third budgetary model input is the indirect marketing budget. This budget contains internal overhead funds allocated for conducting marketing and the initial stages of developing a proposal, before the Bid and Proposal budget is used to fund program managers and engineers. It is a complement to the Bid and Proposal budget, required before the bid and proposal budget is used to further develop a proposal.

³³ These budget categories (discretionary budget, bid and proposal budget, and the IRAD budget) are common to all Department of Defense contractors, and are specified by Federal Acquisition Regulations (see <http://www.arnet.gov>). Using these inputs increases the ability of the simulation model to be easily transferred to other enterprise in the same sector.

³⁴ In some cases, TechSys may decide to increase the Discretionary Budget by reinvesting from its profits.

The fifth and final model input identified was the headcount devoted to new business pursuit and capture, including staff in business development, program management, and engineering available to work on proposals at each operating unit. The headcount is not fully independent from the Discretionary budget and the indirect marketing budget, but it is not directly tied to these sources of funds and is treated as a separate model input.

These inputs were established after two iterations with TechSys stakeholders, aimed at identifying inputs that could be tied directly into the execution of the architecture and were more broadly generalizable across TechSys and to other enterprises engaged in federal acquisition.

Figure 6-1 is a notional “black box”-level depiction of the simulation, showing the set of input parameters on the left that are fed into a “black box” with a transfer function T which is a function of the input parameters. The output of the transfer function, synergy and organic profits due new contracts won, is shown on the right of the transfer function, shown as a stacked, cumulative graph.

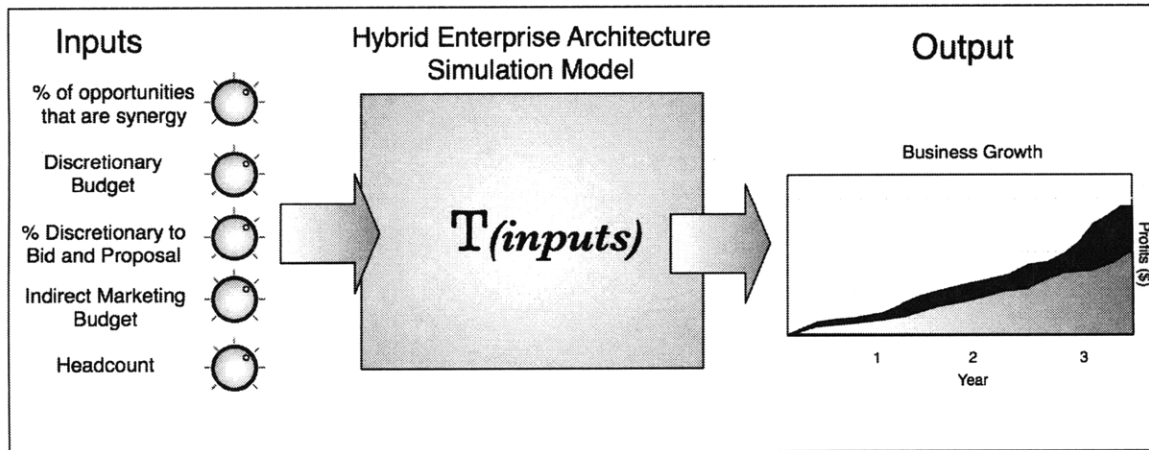


Figure 6-1: Conceptual "black box" level depiction of the TechSys Simulation Model

6.2.3 Time Horizon

The time horizon for the simulation model was set at three years. This was a fairly short time horizon given that the model contains elements of strategy and research that have impacts farther out in time, but the strategic planning process of TechSys operates on a three year cycle. The choice of time horizon should be revisited to understand how varying the time horizon impacts the model's behavior.

6.2.4 Key Structures and Behaviors

The next step of problem articulation is to link the problem to the architectural structures and dynamics of the enterprise by identifying key structures and behaviors in the enterprise architecture that are relevant to the problem. This need not be detailed analysis nor show the future structure of the model; this will come later. The problems that TechSys face balancing its new business pursuit capture are linked very deeply to its enterprise architecture, and are fundamentally dependant on facets of the architecture described by the strategy, process, organization, and knowledge views within the NREAF.

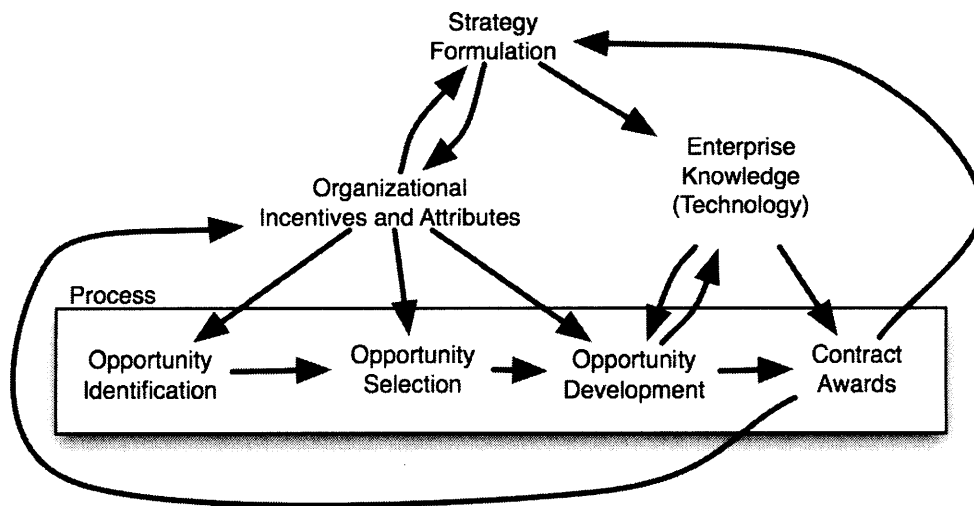


Figure 6-2: A conceptual diagram of relationships between architectural factors related to the pursuit and capture of new business opportunities

Figure 6-2 gives a high-level, notional view of the important factors and structures from the strategy, organization, process, and knowledge views associated with the pursuit and capture of new business opportunities and how they relate to each other. This very simple sketch highlights how the simple, linear process of opportunity development (identification to selection to development to contract award) is part of a larger process across the enterprise with feedback. At the top of Figure 6-2, strategy formulation is shown driving both organizational incentives and characteristics as well as the pursuit of new knowledge (technology). It receives feedback from both the performance of the organization as well as the ultimate performance of the process in the form of awarded contracts. The organization's incentives and attributes drive every stage of opportunity pursuit, from identification to selection and then development. Further, the organizational incentives determine the extent to which the operating unit will participate in developing synergy opportunities and collaborate with others. The organization also receives feedback from the performance of the process in the form of contract awards before it arrives in the form a new strategy. The success of the new business opportunity in the development stage is dependant on the knowledge and capabilities of the enterprise. After the work is completed and if it has been internalized, the organization has then gained experience that will aid future development efforts. That experience also establishes a reputation that increases the likelihood of future contract awards.

6.2.5 Applicability of Hybrid Enterprise Architecture Modeling to the problem

The final step of problem articulation is to ask if the problem could not be more effectively addressed through other more traditional modeling and simulation approaches (or even a non-modeling approach). Resource allocation problems

in enterprises have been commonly modeled through the years. Both System Dynamics and Discrete event simulations have been used to analyze different kinds of resource allocation issues in complex, dynamic environments. These approaches on their own cannot capture dynamics of TechSys's architectural challenges, however.

The problem of achieving synergy between operating units is a problem that crosses many contextual boundaries. In terms of the views of an enterprise architecture framework, the problem touches on the strategy, organization, processes, and knowledge views in a substantial way. Each of these perspectives has its own dynamics: the new business capture process is a probabilistic function that transforms budget resources into enterprise profitability performance outputs; the organization is made up of units with their own incentives making locally rational decisions based on their perception of the world around them; the strategy of the enterprise is shaped by continuous pressures and qualitative relationships. *No single modeling methodology can capture all of these different contexts and their interaction to produce the resulting behavior of the enterprise.* Without understanding the problem from multiple points of view and being able to integrate and understand how those different perspectives interact, it is difficult to get full understanding of the nature and dynamics of the problem. Only a hybrid approach to simulating the enterprise architecture can provide this perspective and analysis.

6.3 STEP 3: FORM A DYNAMIC ARCHITECTURAL HYPOTHESIS

After clearly articulating the problem for the simulation model of TechSys's enterprise architecture, the next step in the hybrid enterprise architecture modeling process is to further scope the boundaries and function of the model by creating a dynamic architectural hypothesis, as described in Section 0. The dynamic architectural hypothesis is based on an initial insight into the problem that explains the dynamics of the problem in terms of the enterprise architecture.

Usually, the enterprise's stakeholders will have some opinions as to the cause of problematic behaviors or barriers to achieving a performance goal. The dynamic architectural hypothesis is a chance to articulate these initial beliefs.

The dynamic architectural hypothesis for the TechSys simulation model was developed with input from the stakeholders, with particular input from the Director of Enterprise Strategy, who was actively leading a team at TechSys working to identify and mitigate barriers to synergistic cooperation between operating units at the time. There was a concern at TechSys that its current growth goals established in its strategy were unattainable using only organic growth; new synergy growth between operating units that would enable expansion into new markets would be necessary. Given recent performance of the enterprise, most felt that the existing architecture was not capable of realizing the necessary synergy growth between the operating units in sufficient quantity to meet its growth goals without the removal of several barriers to synergy, many of which were a part of the enterprise architecture. In light of this environment, the following dynamic architectural hypothesis was proposed for the TechSys simulation model:

Given its current enterprise architecture, TechSys will be unable to meet its business growth and associated profitability goals by utilizing the pool of budgetary resources that are available for pursuing new business growth opportunities. That is, the existing architecture is hypothesized to have a constraining influence on the company's capacity to capture the new business opportunities it has targeted. Hence, the existing architecture must be modified in order for the company to take advantage of the synergistic business growth opportunities facing its various business units in order to achieve the new company-wide business growth goals.

The motivating hypothesis or central strategic question posed above can be evaluated by running a range of feasible inputs (limited budgetary resources) through the simulation model and observing if any combination of inputs can cause the expected value of TechSys's business growth or profits to meet its established goals. If the model can be shown not to result in the new business growth or profitability targets, under any combination of the available types of budgetary resources and how they are allocated, then the maintained hypothesis (i.e., the current enterprise architecture has a constraining influence on the achievement of the targeted business goals) cannot be rejected. If the outputs of simulation model can meet TechSys's goals for some combination of inputs, TechSys could use the model to investigate investment and management strategies with the existing architecture. If not, the model could be used to investigate the effect of modifying the architecture to achieve greater growth potential.

6.4 STEP 4: IDENTIFY THE APPLICABLE VIEWS FROM THE ENTERPRISE ARCHITECTURE

The next step in the process of developing the simulation of TechSys's enterprise architecture is to formally identify the areas of the enterprise architecture that have an effect on the enterprise's ability to pursue and capture new business. The Nightingale-Rhodes Enterprise Architecture Framework identifies eight views of the enterprise, but not all views are equally represented, and some, such as the services view, may not be represented at all in the simulation model.

In this step of the modeling process, each view from the NREAF was considered for inclusion. The most involved views in the simulation model, such as process, organization and strategy, would require a full sub-model in the finished hybrid simulation. Other views, such as knowledge or information technology, had an influence on the problem under study, but the impact of the view could be captured using a few high-level variables within the model, rather than requiring a

fully developed sub-model with its own structure and behaviors. Other views may not have had such little impact on the problem under study that they would not need to be included in the hybrid model. The following sections review each view, and review the factors that determined how the view would be included in the hybrid simulation model.

Views included in the simulation as sub-models:

- *Strategy/Finance View:* This view is critical to the model. Key components of this view are financial allocation of resources between the operating units, pressure to enter into new markets, monitoring performance to strategic plans, and the strategic planning cycle. This view should be developed into a sub-model in the hybrid structure.
- *Organization View:* This is also a critical view, because the organizational structure, incentives and communication pathways play a major role in determining the extent to which operating units are willing to collaborate on synergy opportunities. The primary components of the organizational view of the architecture are descriptions of the operating unit level and the division level. The drivers of OU behavior can be adequately captured using an aggregate focus on the OU, so a more detailed model of the incentives and structures of groups or individuals is not likely needed in this model. If during model testing this is called into doubt, the organizational boundaries can be pushed to a more granular level. This view should be developed into a sub-model in the hybrid structure.
- *Process View:* The process view can be seen as the heart of the model, as the simulation at its core seeks to explore the performance of the new business pursuit and capture process. The key processes used in this model are new business opportunity identification, selection, pursuit, and capture. Additionally, the internal research and development process is included. Other lifecycle processes, such as production, engineering or

support are not included. This view should be developed into a sub-model in the hybrid structure.

Views included in the simulation as key high level variables:

- *Knowledge View:* Knowledge plays a critical role in the model, but it is influenced by other views and does not have internal structure and behaviors that is useful for the problem at hand. An organization only wins new business if it can deliver the right knowledge and capabilities for each proposal. This knowledge is developed both through investment in internal research and development and through experience with related programs. This knowledge component can be captured using model variables that indicate the maturity of the knowledge of the enterprise in key areas. It does not require a sub-model that has its own structures and behaviors.
- *External/Policy View:* While this model is primarily internally focused on TechSys without major external dynamics, there are external/policy considerations that impact the problem, such as government accounting rules that limit collaboration between operating units and the competitiveness of different markets, as well as the Department of Defense acquisition budget. These considerations must be included in the model, but do not rise to the same level of detail required by other views.
- *Information Technology View:* One of the keys to synergy growth between operating units is effective communication and interoperability. While there are many potential information technology interfaces between operating units that must ultimately be considered (such as database designs, tool choices, and other physical connection interfaces between operating units), these interface details are necessary at the operational level of the architecture, and need not be modeled in detailed to achieve the insights desired by this hybrid simulation model. Ultimately, these interfaces should be modeled in detail to implement greater interoperability, but in

this model interoperability between operating units can be modeled using aggregated variables describing the strength of the relationship.

Views not included in the simulation model:

- *Product View:* While the goal of winning new business is to produce and sell new products, there are no attributes of the products themselves that are strictly relevant to this problem that are not captured by other views (such as technologies and markets). For this reason, the product view is not used for this model.
- *Services View:* The problem does not touch on any elements of the service elements of TechSys' enterprise. While an operating unit may bid on providing a service instead of a product, this is irrelevant in the model.

6.5 STEP 5: MATCH THE VIEWS WITH SIMULATION METHODOLOGIES

The fifth step of the modeling process is to determine which of the simulation approaches described in Chapter 3 should be used to model the structures and behaviors of each view included in the model. The modeler must decide which of the available simulation methodologies is most appropriate to describe and analyze the behaviors in each view, keeping in mind the strengths and weaknesses of each approach. This selection can be a difficult process, as multiple approaches could potentially be used to model the behaviors of a single view. The next sections will go through the sub-models identified in the previous section and describe how each was matched with a simulation methodology. After describing how the sub-models will be simulated, the modeling structures corresponding to the knowledge, information technology, and external/policy views will be described.

6.5.1 Strategy/Finance Sub-model

The strategy/finance sub-model could potentially be simulated using any of the available simulation approaches. Given that the key structures within the sub-model are fixed processes with flows of money (which is homogeneous), implementing the sub-model as an agent based model would not be the first choice. This leaves system dynamics or discrete-event modeling as potential methodologies. From a discrete-event point of view, financial activities could be thought of as discrete bins of money that are allocated and updated on a monthly cycle, and strategy planning activities could be modeled with software routines called according to a calendar in the model. Specific financial and strategic targets could be established in the discrete event model that could provide a control feedback loop in the larger hybrid model.

The second approach to modeling the strategy/finance view, employing system dynamics, would take a continuous time perspective. Finances could be considered stocks and flows of money that could be influenced through work and billing rates determined by the operating units. Strategy would be modeled as a balancing system dynamics feedback loop that seeks a desired target state (growth goals), and directs action in the hybrid model in response to a gap between the current state and the desired state.

Realistically, either modeling approach could be employed with likely success in this particular hybrid model, as both allow for controlled feedback following a defined schedule and accounting and allocation of resources, which are the primary dynamics of the strategy/finance views. For purposes of this case study to establish proof of concept, however, employing system dynamics would be more illustrative, since it would not be employed elsewhere in the model, and as the reader will discover, discrete event simulation is heavily used for the process sub-model. Additionally, the system dynamics sub-model would be slightly faster to implement.

6.5.2 Organizational Sub-model

The key behaviors in the organizational sub-model are decisions made by each operating unit as it selects projects fund, based upon which projects provide the most benefit to that operating unit. This behavior closely describes the primary strength of agent-based models, and this behavior cannot be simulated using other approaches without significant customization. An agent-based modeling approach can be used to capture the decision making of local operating units and how these decisions impact the other operating units and TechSys as a whole. Such an approach treats each operating unit and the division itself as an agent with its own decision-making rules. The agents would collaborate amongst themselves given the rules in their schema for cooperation, trying to maximize their own internal value functions, just as the general manager at an operating unit has ultimate profit and loss responsibility for his operating unit. This way of modeling is in line with the way most people think about organizational issues; organizations interact, and do so according to their own incentives. The agent in the model can have internal functions and attributes that describe its characteristics in relation to other agents. In this way, operating units can be compared to one another along organizational dimensions such as process compatibility or frequency of interaction.

6.5.3 Process Sub-model

The process model could technically be modeled with any of the approaches, however using system dynamics or agent based modeling would be difficult and would not use the strengths of either approach. An established selection process with decisions at each step does not lend itself well to the causal loop structure of system dynamics. Further, individual project proposals would need to be aggregated in any processes, making it difficult to identify bottlenecks and operational dynamics. An agent-based model might do better than system dynamics, but the processes do not require the intelligence or flexibility of agents. When compared to discrete event simulation, using an agent-based model would

require more effort and would be more difficult to explain to stakeholders reviewing the model.

Discrete event simulation is the best available approach to model TechSys’s processes. TechSys’s business pursuit processes execute a defined, logical process on a schedule, with decision points (review gates) at intervals. Discrete resources (engineers, program managers, business development specialists) are assigned to discrete, unique business opportunities as they flow through the process from identification to selection and later development. The attributes and handling of these opportunities can also be treated as probabilistic. All of these qualities point to the use of discrete-event modeling as the preferred approach to simulating TechSys’ process dynamics related to business opportunity pursuit and capture.

Table 6-1 provides a summary of the modeling approaches considered for each sub-model. An “x” indicates a simulation approach that was considered for that sub-model, and a bold “X” indicates the approach chosen.

Table 6-1: Selection of simulation approaches for the sub-models

	Agent based model	Discrete Event Model	System Dynamics
Strategy/Finance sub-model	x	x	X
Organizational sub-model	X		
Process sub- model	x	X	x

6.5.4 Modeling other Views with Variables

Not every enterprise architecture view requires a full sub-model to understand the behavior of the enterprise architecture with respect to the pursuit and capture of new business opportunities. The next three sections will describe how the knowledge, external/policy and information technology views can be modeled using variables and rules in the hybrid model that exist external to the sub-models and interact with the sub-models.

Knowledge view modeling

The knowledge dynamics associated with TechSys' business growth are primarily associated with how each Operating Unit plans for its technical capabilities and ensures that it will have the right core competencies to be competitive in the markets that it desires. Here, there is some potential overlap with the strategy sub-model and perhaps the process sub-model. Each operating unit separately maintains its own knowledge register (a list of core technologies and their maturity) and executes IRAD processes to increase the maturity of these technologies. Technology maturity increases their ability to win proposals in those given areas. This dynamic would best be captured with a "technology register," a variable array that can be read by the strategy sub-model and changed by the process sub-model. This technology register would be an interface between the strategy and process sub-models, but would not be a sub-model in its own right

External/Policy modeling

While there are enterprise architecture considerations and structures related to the external/policy view that much be considered as part of the hybrid simulation, they are not dynamics, per se, that must be modeled. The two primary external/policy considerations are (1) external rules imposed on the architecture for how the enterprise is allowed to communicate between operating units, and (2) a list of possible markets, their competitiveness, and their profitability. The first consideration can be incorporated into the model by including the external

rules that limit collaboration as part of the schema of the organizational agents; the second can be included in the model by creating a variable array of potential markets for the operating units, similar to how the technology register in the knowledge view was implemented. While the market array would be an important component in the model that every operating unit would interact with, it would not be a dynamic sub-model of its own, but rather a single global structure that is updated by other sub-models.

Information Technology Modeling

While there are many dynamics associated with information technology at TechSys, given the scope of the model, the only impact that IT has is its effect as an enabling factor in facilitating the collaboration of operating units. From the point of view of the model, it could be viewed as a single variable, "effectiveness of IT," that would be embedded as a characteristic of each operating unit. This variable is a multiplier of communication effectiveness. It would be possible to fully develop an IT model using system dynamics that involves feedback from strategy whereby funds are invested to increase capabilities of IT, given some transfer function that turns money into IT capability. Given the observations of the relative impact of IT on the issue of new business pursuit and capture, TechSys stakeholders unanimously agreed that including a higher fidelity model of IT dynamics was not likely to be valuable. Evaluating the current processes in place, this decision is sound. If future processes are developed that rely more heavily on IT, however, this simplifying assumption must be revisited.

6.6 STEP 6: IDENTIFY BOUNDARIES AND INTERFACES BETWEEN THE VIEWS/SUB-MODELS

After choosing the general methodologies to model the dynamics of the TechSys enterprise architecture, the next step is to identify the boundaries and interfaces of the sub-models in the hybrid simulation. This is a critical step, as the

explanatory power of a hybrid architectural simulation comes from the ability of each sub-model to accurately depict the dynamics of its associated architectural view and then interact with the other sub-models to produce enterprise-level dynamics. As developed in Section 4.7.6, the identification of boundaries and interfaces can be separated into two steps: (1) creating a simple logical diagram of the interaction of the views with active stakeholder involvement; and (2) creating boundary model charts for each view.

6.6.1 *Creating a Logical Interaction Diagram*

A major benefit of using the NREAF to build TechSys' enterprise architecture is that the framework has already identified the generic logical boundaries and interfaces between its views. Given the generic identification of the view boundaries and interactions developed by NREAF, the next step of the process is to then adapt them to the particular case of TechSys' new business pursuit and capture dynamics with active feedback from TechSys stakeholders.

The logical interaction diagram for the TechSys model is shown in Figure 6-3. This is an adaptation of the general interaction diagram from NREAF to the specifics of the TechSys model. The strategy, organization, and process views are shown as sub-models (boxes), while the other included views are shown as variables and constrains (rounded shapes).

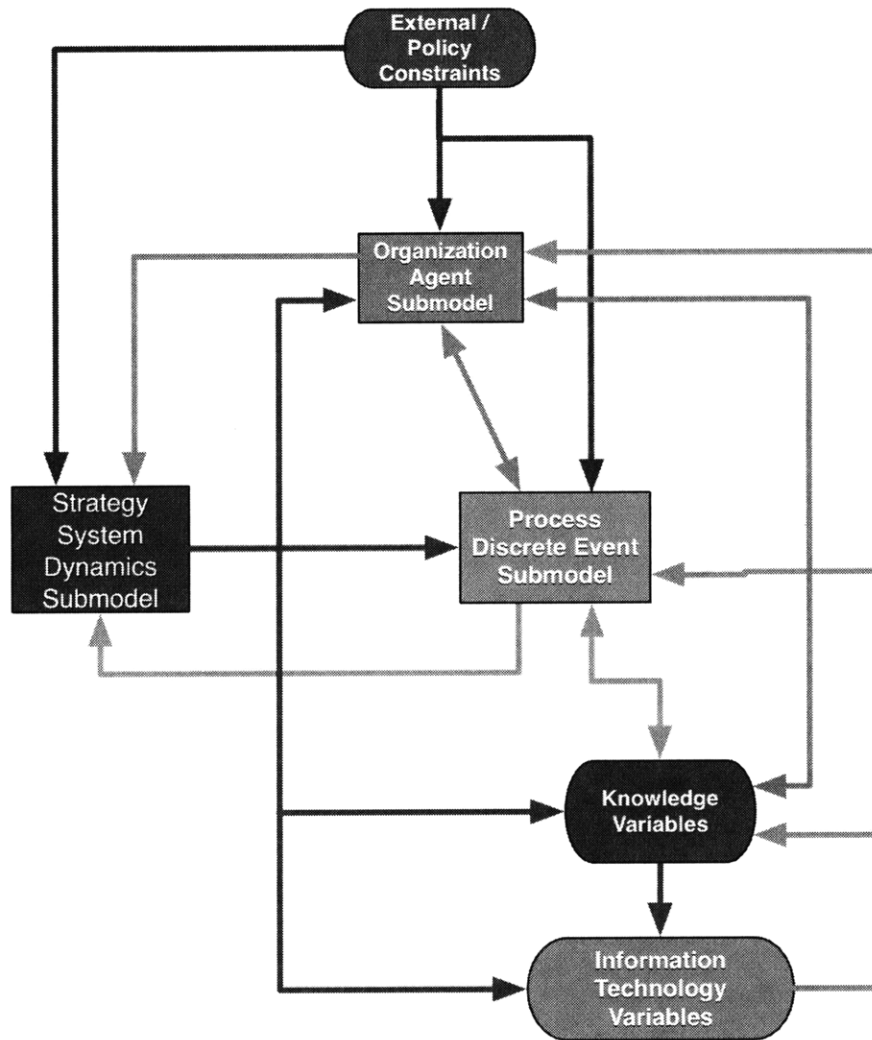


Figure 6-3: Logical Interaction diagram of the TechSys Enterprise Architecture Simulation Model, based on the NREAF Logical Interaction Diagram

Figure 6-3 was the output of discussions with TechSys stakeholders aimed at adapting the generic NREAF interaction diagram to the specifics of TechSys' new business pursuit and capture dynamics. First, the product and services views were removed from the diagram because they are not included as part of the simulation (Section 6.4). Next, each arrow in the generic diagram was reviewed to determine if an interaction between the two views should exist in the TechSys model. At this stage only the presence of an interaction was noted; a detailed description of the interface with a unique variable is a later step. After completing this process a final check was performed to determine if there were

any interactions present in the dynamics of TechSys' enterprise architecture that were not shown in the generic NREAF logical interaction diagram.

The logical interface diagram in Figure 6-3 serves as a useful tool when analyzing TechSys' new business pursuit and capture dynamics, because it forces the diagram user to consider the interactions that lay across logical boundaries instead of focusing on the dynamics within a particular view. For example, it may be tempting for TechSys to only consider new business pursuit and capture from the perspective of its processes or only its strategy development. The logical interaction diagram forces its user, either as part of a model building process or as part of a mental analysis activity, to explicitly consider the interaction between the different contextual views. This encourages the user to look at the problem from multiple perspectives (strategy view vs. process view), or it may remind the user to consider the impact of interactions with other views that may be otherwise overlooked without attention specifically called to it.

The process of creating the logical interaction diagram can also highlight weaknesses in the current enterprise architecture. While working with TechSys stakeholders to identify architectural interactions between the views, for example, it was discovered that the current TechSys enterprise architecture makes no provisions for a key feedback interaction identified by NREAF between the organization view and the strategy view. There is no process or structure in the architecture to assess organizational performance against enterprise strategy; this is currently done in an *ad hoc* fashion and is dependant on perceptive leadership. Some stakeholders felt that this lack of a pathway for feedback in the architecture was a weakness that should be addressed in the future.

This discussion, arising from the creation of a simple diagram, demonstrates that following rigorous processes in hybrid model creation is valuable. The discussion surrounding the feedback interaction between organizational

performance and strategic assessment led to an insight highlighting a weakness in the current enterprise architecture. It also generated data that was used as an input when creating the model, and very importantly, it was a source of stakeholder involvement and buy-in. By using a very simple diagram with a group of stakeholders to walk through these issues, stakeholders were able to participate in discussions centered on the enterprise architecture.

In an early iteration of this process with enterprise stakeholders, a detailed version of a logical interaction diagram with detailed variables was presented, but the conversations became bogged down in details before the higher-level function could be determined. Focus was taken off of the interactions between the views and became centered on specifics of process or strategy to the detriment of understanding the cross-view interactions and dynamics. In addition to becoming bogged down with details and slowing progress, stakeholders did not see the usefulness of how the Nightingale and Rhodes Enterprise Architecture Framework could be used to partition the enterprise, thus causing the framework to lose its intended value. In response to these challenges, the process was revisited with the more simple form of the logical interaction diagram shown above with the desired results.

While the logical interaction diagram is a useful tool in creating the hybrid simulation, its power lies mostly in the insight and buy-in that arises as a by-product of its creation with enterprise stakeholders. It is of only limited help when actually creating the model; it does not contain any details of the boundaries of the views or explicitly identify interfaces between the views. Before a hybrid simulation model can be created in software, the logical interaction chart must be extended using partial boundary model charts that explicitly identify the boundaries of the views and specific variables that serve as interfaces between the views, complementing the logical interaction diagram's shortcomings.

6.6.2 *Boundary Model Charts*

Boundary model charts extend the high-level overview given by the logical interaction diagram by explicitly listing which dynamics and variables should be internal to the view, which are not included in the view, and what variables serve as outputs and inputs to the sub-model of the view. In the case of the TechSys hybrid simulation, there are three views that were developed into sub-models (strategy, organization, process). For each of these sub-models, a boundary model chart was developed, as presented in Section 4.7.6. In each of the charts, the key input and output interface variables were identified, as well as the endogenous dynamics that were included and any variables of dynamics that could have been included but were intentionally excluded.

As opposed to the logical interaction diagram, the boundary model chart contains details that are more specific in nature and harder to discuss with a diverse collection of stakeholders. For this reason, the boundary model charts were developed with feedback from specific stakeholders who had insight into the dynamics of a given view. These were developed with input from the documented enterprise architecture, with follow-up interviews scheduled to address gaps in understanding.

Table 6-2: Model Boundary Chart of the Strategy/Finance View sub-model

Endogenous	Outputs	Exogenous (Inputs)	Excluded
-Division of budget between operating units -Strategy Planning Cycle, with delays	-New technologies to pursue -New Markets to pursue -Age of Strategic plan	-Top Level Model Financial Parameters -Financial performance of operating units -Markets (External) -Technology Roadmap (Knowledge)	-Active selection of new markets -New strategy definition -Leadership quality -capital expenses,

Table 6-3: Model Boundary Chart for the Organizational View sub-model

Endogenous	Outputs	Exogenous (Inputs)	Excluded
-Physical connectivity -Logical connectivity -Number of shared/similar customers -Shared processes -Shared infrastructure -Number of inter-OU personal relationships	- Operating unit revenue, profit	-Corporate IT expenditures -Strength of Corporate communications -Personnel Mobility between OUs (hr/mo) -Financial teaming incentives -DCAA rules for collaboration	-Leadership quality -roles of individuals in the OU -a specific agent for R&D

Table 6-4: Model Boundary Chart for the Process View sub-model

Endogenous	Outputs	Exogenous (Inputs)	Excluded
-Opportunity entities (see Table 6-5)	-Revenue from contract awards	-IT effectiveness	-Details of proposal development
-Probability of passing each gate review		-Strength of Corporate communications	-Red Team reviews
-decision making criteria for each gate, selection point		-Personnel Mobility between OUs (hr/mo)	- Opportunity level assignment
-Selection criteria for opportunity register		-Financial teaming (dis)incentives	-Gate deliverables (such as graphics, reports, etc.)
		-Headcounts	-creation of actual "bluesheets"
		-input budgets	-show tool use
		-Historical OU process performance	(design scorecard, Pugh Matrix, etc.)
		-Markets Register	
		-Technology Register	
		-Strategy maturity	-any details of R&D process

Defining boundaries and interfaces is often cited as one of the most difficult steps in the creation of any model, and this also proved true when developing the hybrid simulation model of TechSys. There were at least three iterations where boundaries and interfaces were revisited due to inconsistencies or logical disconnects discovered later in the process. Iteration with boundary identification should be anticipated, especially in conjunction with the next step in the process, creating model diagrams.

6.7 STEP 7: CREATE SUB-MODEL AND TOP-LEVEL MODEL DIAGRAMS

With an assessment of boundaries and logical interfaces in place, the next step is to lay out the structural form of the hybrid simulation using sub-model and top-

level model diagrams. These diagrams are abstractions of the structure and function of each sub-model, as well as how they all fit together and interact to form a cohesive hybrid model. It is in this step that the overall hybrid structure of TechSys' simulation comes into focus, leading to the implementation of the model itself.

6.7.1 The Sub-Model Diagrams

A sub-model diagram is important because it is used to develop the actual sub-model used in the simulation as well as a communication aid when working with stakeholders during model validation. For this reason, the sub-model diagrams were created with feedback from TechSys stakeholders who had a broad knowledge of the enterprise architecture.

Strategy Sub-Model Diagram

Previous steps identified three sub-models to be used in the TechSys hybrid simulation model: strategy, process, and organization. The internal variables, as well as input and output interface variables for these sub-models, were defined in the previous step, leaving the sub-model diagram to extend those definitions by graphically depicting the relationship between the variables within each sub-model. Where possible, each sub-model is captured using a diagramming method that is typically associated with the modeling methodology that the sub-model employs. For example, Figure 6-4 shows the strategy sub-model diagram developed using the partial boundary model chart from Table 6-1. Since the strategy view employs system dynamics, the sub-model chart is shown as a causal loop diagram, an approach that is traditionally used by system dynamicists to show the structure of a system dynamics model. It does not yet show the quantitative relationships between the variables, but it clearly shows the input variables, the output variables, and the direction of the relationship between variables internal to the sub-model.

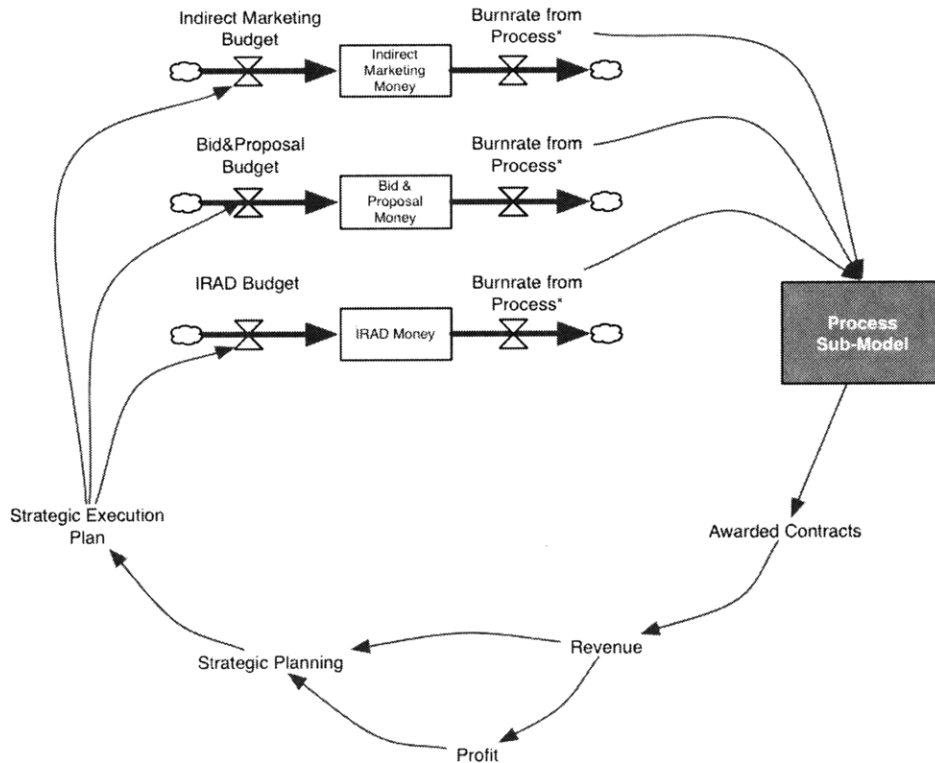


Figure 6-4: Sub-model Diagram of the Strategy Sub-model

Organizational Sub-Model

Figure 6-4: Sub-model Diagram of the Strategy Sub-model shows the sub-model diagram developed for the organizational view, which employs an agent-based methodology. Unfortunately, there is no established method for graphically capturing the structure of an agent-based model, as agent-based models can differ drastically depending on their implementation. Agent-based models often have no initial structure, but instead develop a structure over the course of model execution. In the case of TechSys' organizational sub-model, the dynamics between agents (operating units) are largely driven by the rules that define how operating units interact with each other and with the division. For this reason, the organization sub-model is shown as an organizational hierarchy with established pathways for communication, and each agent behaves in accordance with its internal schema along these pathways.

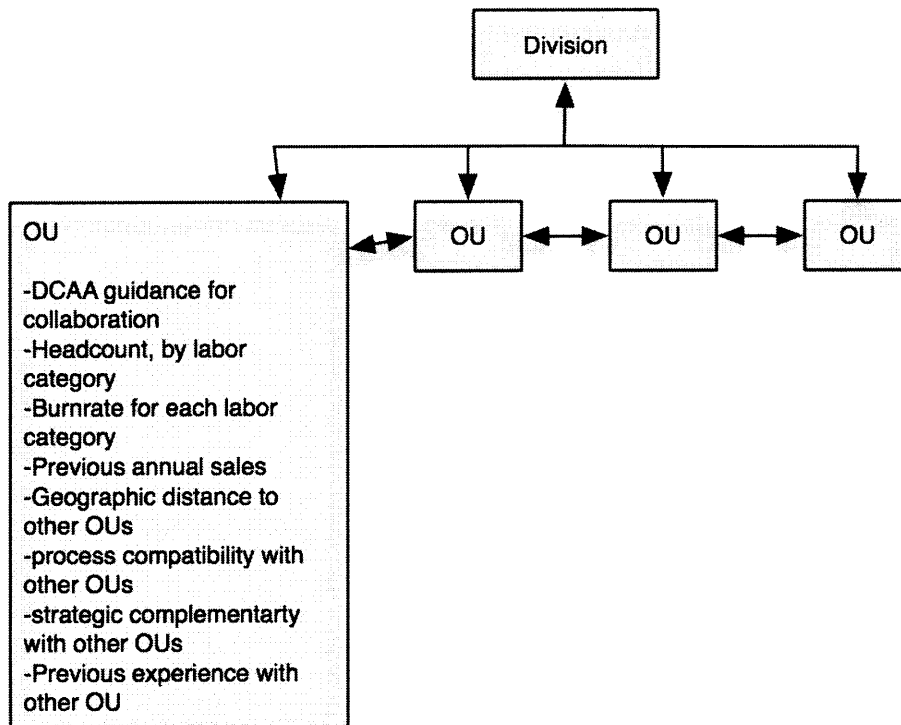


Figure 6-5: Agent structure for the organizational sub-model

Each agent in the sub-model is depicted as a rectangle in Figure 6-5. The double-headed arrows indicate that all communication pathways are bidirectional; an operating unit can communicate directly with other operating unit without having to go through the division. The agent on the far left is enlarged to list the its defining attributes related to the organization view of the enterprise. Unfortunately, because the agents in the sub-model will use their attributes as inputs into computer algorithms (schema) that governs their decisions and interactions, it is difficult to depict the relationship of the attributes or the agents graphically, other than to show that they are connected.

While the structure of each operating unit agent shown in Figure 6-5 is identical, the values of their defining attributes are different for each agent. This way, each agent represents a unique, real-world operating unit that is part of TechSys, yet the structure of the model itself remains flexible. Because all agents share the

same organizational architecture (for at least the purpose of this model), the model can be easily extended to consider the effect of an acquisition or divestment on the part of TechSys by dropping a new agent into the hybrid model and specifying its initial variable values.

One assumption that was made as a modeling convenience was to treat the research and development unit at TechSys as its own operating unit with high connectivity to all other operating units. In the actual architecture, the research and development unit is a matrixed organization that exists at both the operating unit and the division level. At the operating unit level, this unit conducts research and development and plans out technology needs for that operating unit. At the division level, however, the unit is treated much like its own operating unit, with its own resources that must be managed. At the division level, research and development is conducted that impacts multiple operating units, or has major strategic implications. After initially beginning to develop a separate class of agent to represent this division research and development unit, TechSys stakeholders determined that it would be simpler to treat the unit as a instance of a regular operating unit, with attributes that would capture its size, resources, and ability to easily collaborate across operating units and conduct strategically focused research with wide impact. This assumption and simplification should be revisited during model testing. If it is deemed inadequate, the research and development unit can be remodeled using a dedicated agent class with communication pathways similar to the top-level division agent.

Process Sub-Model Diagram

Figure 6-6 shows the sub-model diagram for the process view. The process sub-model diagram, while large, is a fairly straightforward representation of the key processes that affect TechSys' ability to attract new business, including opportunity creation, selection, development, and internal research and development. These processes are captured using standard flow chart notation, with process boxes and decision steps shown as diamonds. The decision steps

are probabilistic. The process can modify internal variables, such as burnrates or resource allocations. Where this is done, the variables that are used are listed below the process step in the diagram in the figure.

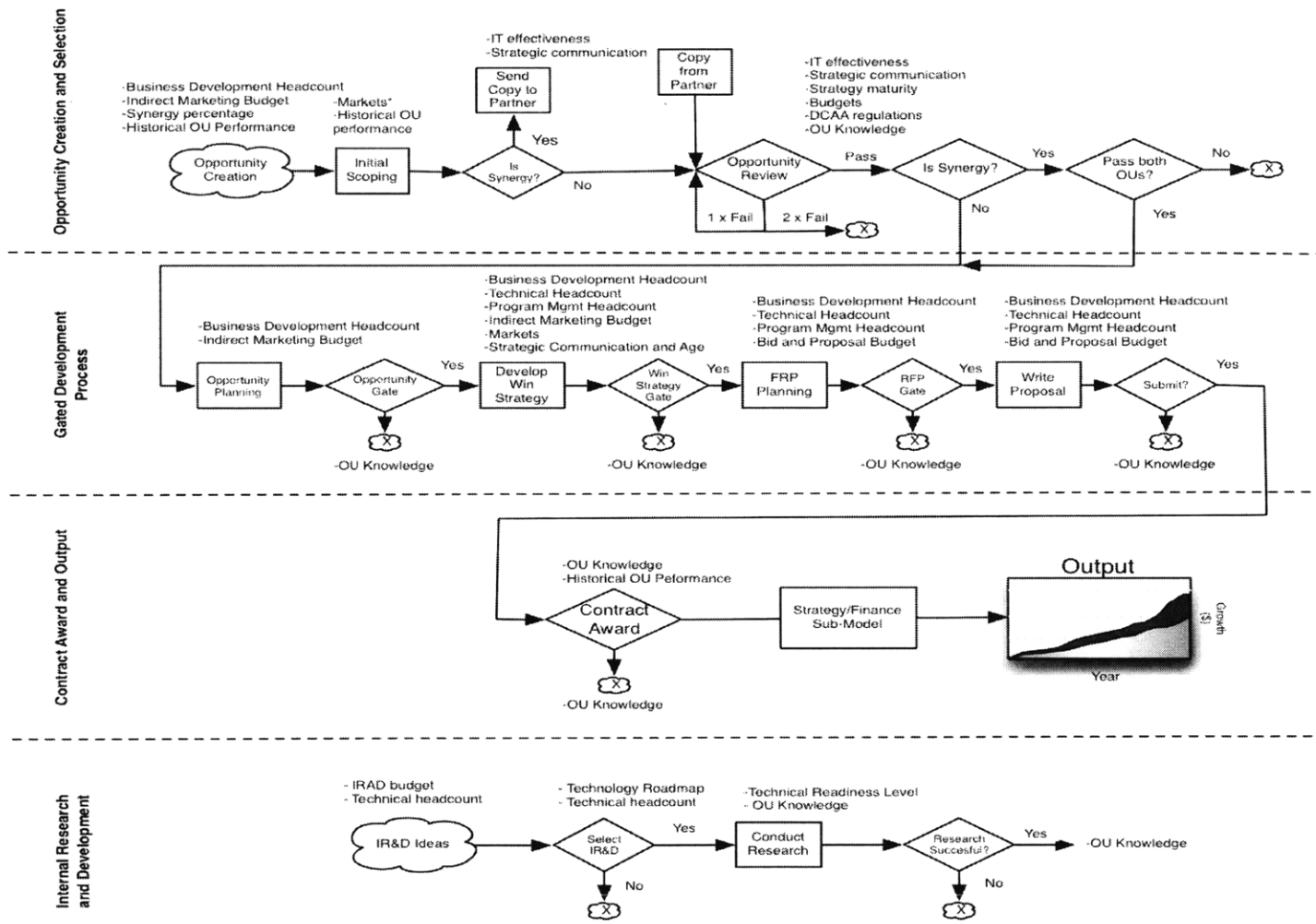


Figure 6-6: The sub-model diagram of the process view

The process diagrams in Figure 6-6 are not sufficiently detailed to re-create the actual processes, but rather represent the fundamental architecture of the processes to understand the flow and major decision points. During development of this sub-model, much more process architecture data was collected than is shown in Figure 6-6. However, after iterating back and forth between creating the sub-model diagrams, creating the sub-models, and then running and evaluating the sub-models, not all of the detail that was collected was necessary to replicate the enterprise-level dynamics over the range of values considered. As a result, process steps that did not have significant impact on the ultimate performance of the process were removed from the diagram in keeping with the maxim that models should be kept as simple as possible while still replicating the intended behavior. Steps that were intentionally not included in the sub-model diagram included additional review cycles, intermediate work products during each stage, and highly specific tasks that are not inputs for later steps (such as graphics development for the proposal) that while necessary to schedule from a program managers point of view, are not necessarily part of the architecture of the process. These excluded process steps/variables are listed in Table 6-3 under the “excluded” category.

6.7.2 Hybrid Model Diagram

The top-level diagram is a representation of how all of the sub-models (as well as the variables capturing the other NREAF views, such as knowledge) will fit together in the hybrid simulation model. While the hybrid model diagram is presented here sequentially after the sub-model diagrams, it was developed iteratively with the sub-model diagrams. To begin the iteration, a rough sketch of each sub-model was made, followed by a sketch of how each sub-model was intended to fit with the others. As mismatches and gaps in the hybrid model diagram were identified, the sub-models were revisited to make corrections or additions.

In the case of TechSys, the primary dynamics identified during problem articulation are the result of each operating unit somewhat independently executing its own processes in accordance with TechSys strategy; the performance of the operating units taken together produce the outputs of the hybrid model. This structure lends itself to the use of the operating unit as the hybrid model's unit of analysis. In this arrangement of sub-model relationships, each operating unit would "own" its own process sub-model and strategy sub-model, as well as other variables and rules used to capture elements of the knowledge, IT, and external/policy views. This forms the basis of an "operating unit-centric" topology³⁵.

The operating unit-centric topology is not the only topology that could be used to capture TechSys' enterprise dynamics; several others are possible. It could be possible, for example, to have a single, central process sub-model that takes inputs from multiple operating units. This topology was also considered for the TechSys hybrid simulation. When deciding between the two possible topologies that could both capture TechSys' enterprise behavior, the operating unit-centric approach was chosen because it would make most sense to TechSys stakeholders. While TechSys is a process-centric enterprise, its stakeholders think about the processes in terms of the operating units and the execution of the operating units. From a stakeholder perspective, the hybrid model would be more realistic and easier to understand if it was viewed from the operating unit-centric perspective. It would also have been more difficult from a programming perspective to create a single process sub-model that could accurately reflect the process performance of all operating units in a single sub-model.

In the operating unit-centric topology, each operating unit executes its own processes and strategy. For this reason each operating unit agent in the hybrid model encapsulates its own copy of the process sub-model and strategy sub-

³⁵ Here, the term topology is used in a computer science and engineering context to refer to the pattern of connecting system components.

model, as well as any variables associated with the knowledge view, the external/policy view, and the IT view. Figure 6-7 is a high-level “box” diagram of how a single operating unit would encapsulate these sub-models and variables.

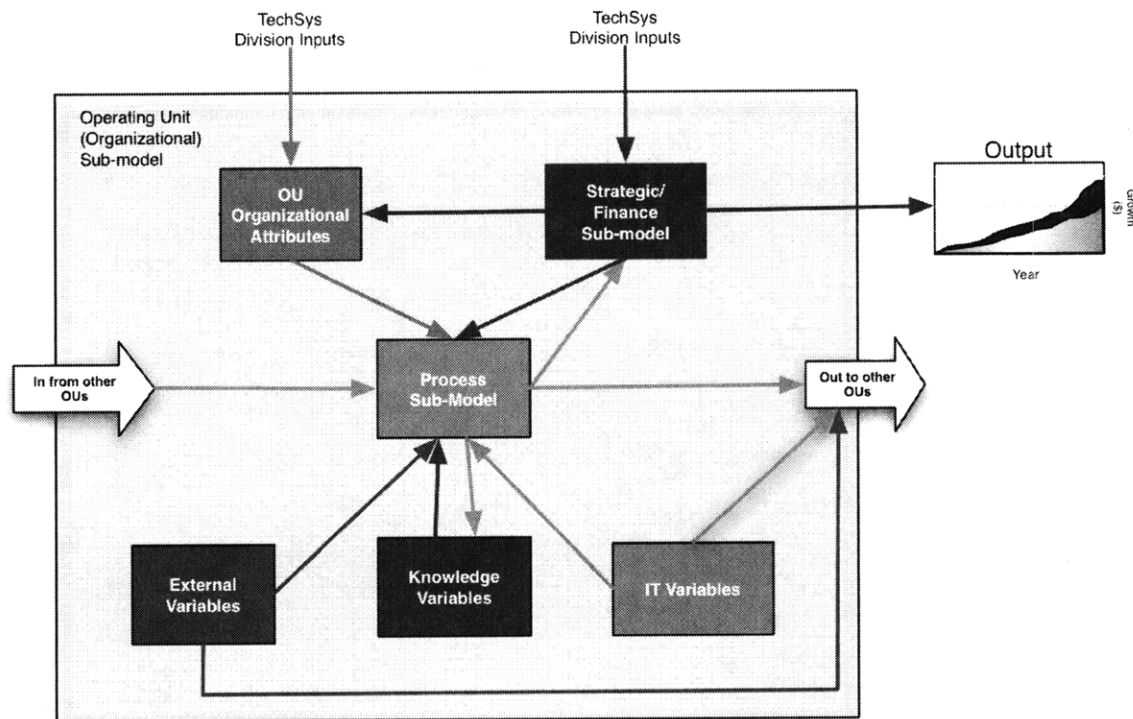


Figure 6-7: A high-level structural diagram of an operating unit with its encapsulated sub-models and variables

After identifying how the sub-models interact at a macro-level within the operating unit, the next step is to increase the resolution of the diagram to show how the key features of the sub-models interact. This is especially important for the process sub-model, which is highly dependant on input from other sub-models and variables at its process steps and decision points (as shown in Table 6-2).

TechSys' operating unit sub-model diagram was increased in resolution by including details from the sub-model diagrams in the operating unit diagram, as shown in Figure 6-8. The objective of doing this was to explicitly identify the interfaces and structures of the model in such a way that the model could begin to be built. While the diagram does not have all of the detail that the final model

would contain, it does capture the primary elements and interactions between major components of the sub-models.

With the operating unit diagrams completed, the final step to create the hybrid model diagram is to link the operating unit agents together as part of the larger TechSys enterprise, showing the model inputs and outputs. For TechSys' simulation model, this shows all of the operating units linked together via the TechSys divisional management organization, with inputs being fed into division management and passed down to the operating units, and financial performance passed back up to the division. An abstraction of the internal process and strategy sub-models of each operating unit is shown as well to serve as a reminder of the internal operation of each operating unit agent. For ease of display, only four operating units are shown, rather than the full complement of seven. See Figure 6-9.

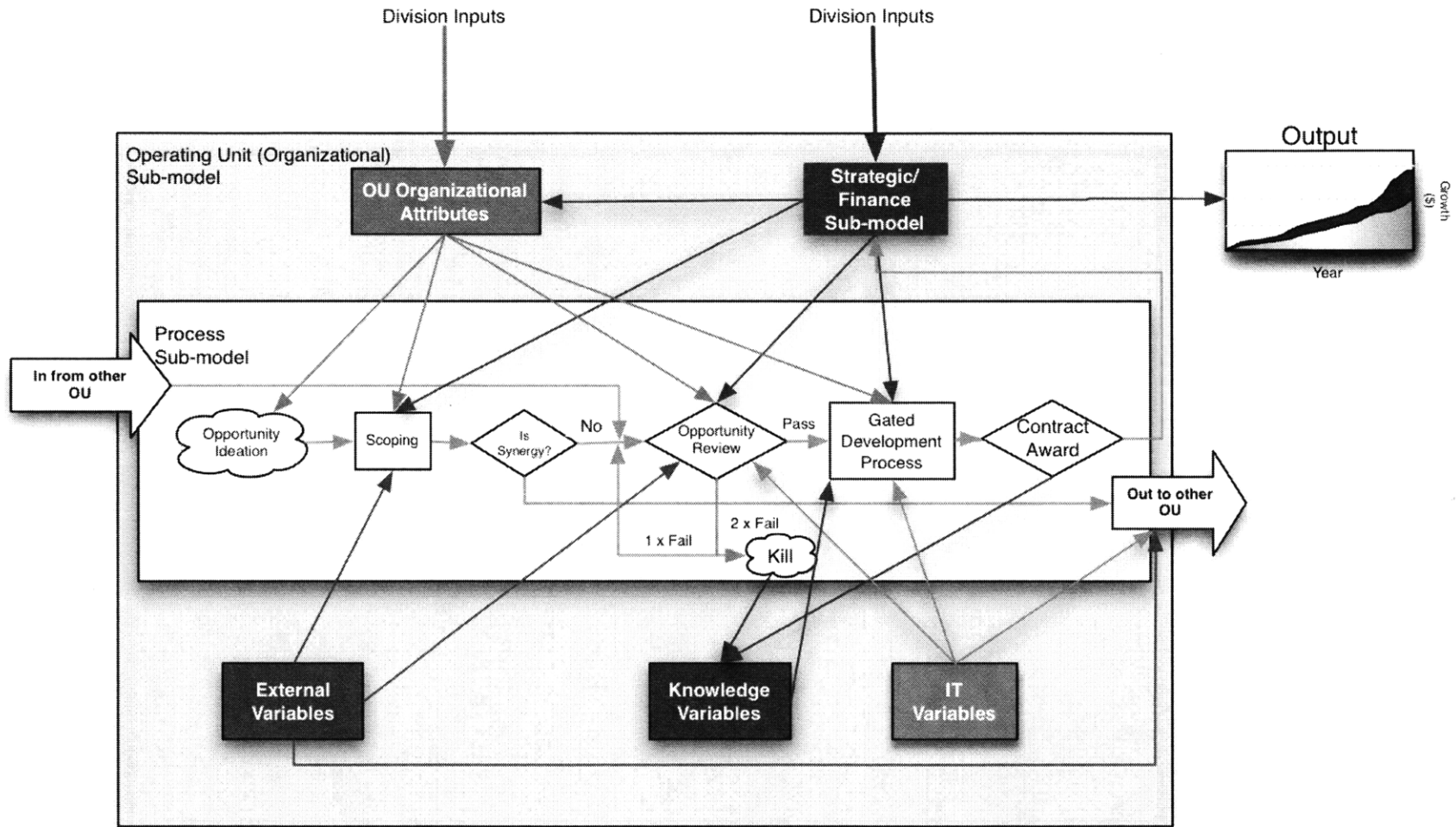


Figure 6-8: Organizational sub-model diagram of an operating unit

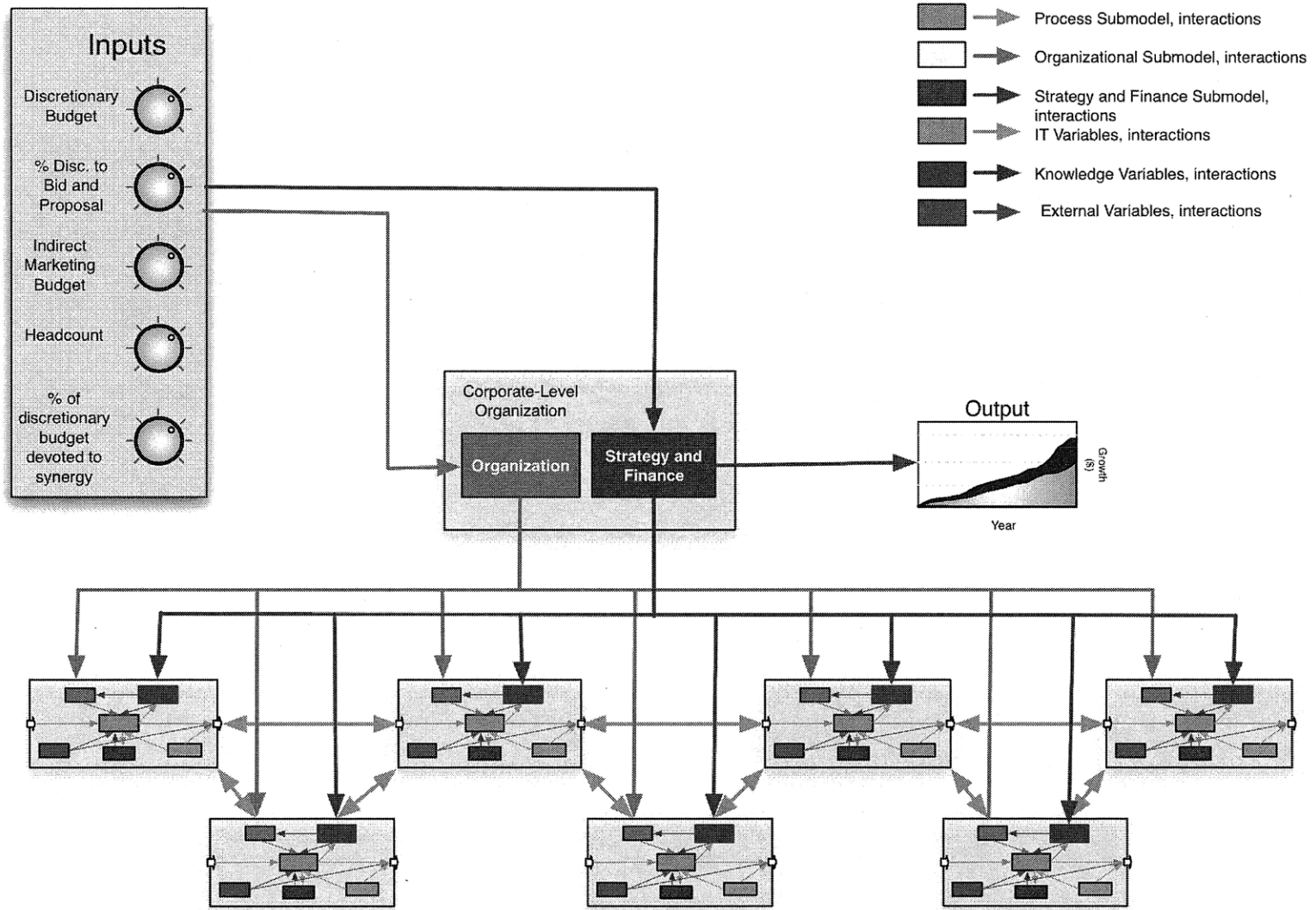


Figure 6-9: Top-level diagram, showing input and output parameter

6.8 STEP 8: IMPLEMENT THE HYBRID SIMULATION MODEL

The steps thus far in the hybrid modeling process prepared the modeler with the necessary tools and structure for the next step: implementation of the hybrid simulation model. While most of the structure, variables, inputs, outputs and interactions associated with the TechSys hybrid simulation had been identified, work remained to quantify these variables and their relationships, collect historical performance data, make qualitative assessments, and implement the simulation in software.

Lessons learned

Without the planning and preparation work of the previous steps, model implementation would have been far more difficult. Without this level of process rigor, model construction could easily veer away from its enterprise architecture-based, hybrid structure towards a more standard, single methodology/context approach. It was difficult to keep the structures and dynamics of different modeling methodologies separated while implementing the simulation; the problem articulation, boundary charts and various diagrams proved indispensable during model implementation and served as reference points during model creation when development efforts began to stray from their intended course. Programmers and enterprise managers alike do not think in terms of hybrid structures, so it was difficult to create the model without constantly referring back to the guidance. The temptation was strong to fall into a single existing perspective when coding the model.

Acquiring reliable enterprise data from a number of sources from an organization not accustomed to capturing it, developing the software, and designing a stochastic approach to the process sub-model all proved challenging to implement. The process of modeling something as broad as the enterprise architecture forces the modeler to identify and relate people and resources across the enterprise that may not be aware of each other, but who have an

impact on each other, such as those in finance, contracts, and engineering. Fortunately, most of the hurdles to implementation are due to the challenges of implementing such a model for the first time; subsequent models would undoubtedly be much faster to implement.

6.8.1 Acquiring Data

The first and perhaps most difficult step of model implementation was acquiring data to back up all of the variables, relationships, initial conditions and rules identified in the previous steps. Much of this data were initially identified when stakeholders were interviewed as part of the process of documenting the enterprise architecture. The interviewees were often able to provide data directly or were able to identify individuals who had the data. Other data sources were found with the help of supportive and experienced individuals at TechSys.

When collecting data for the simulation, the data's owner would also be interviewed, first to verify the associated piece of the architecture and its associated processes, structure and dynamics, and then to inquire about data and its validity for use in the model. These interviews provided significant opportunity for iteration back to reconsider the enterprise architecture and construction of the sub-models. The sub-model diagrams and process model boundary charts were used as points of discussion were applicable. In several instances, discrepancies were identified between the architecture described by a senior manager and that described by others more directly involved with the operation of the enterprise. In cases of discrepancies, the architecture variant best supported by data was used in the model, and the discrepancy was noted in the model documentation.

In the case of processes, the issue was often not a disagreement regarding the documented process, but rather actual execution of TechSys according to the process. "The problem is often not with our processes; it's that we don't follow

the ones we have” was a refrain heard multiple times. In such cases, the *de facto* process was implemented in the model when possible. One such notable case is the process for how two operating units jointly choose which synergy opportunities pursue; currently there is no formal documented process for doing this, but there is a *de facto* process that is used. This *de facto* process, as it turned out, plays a significant role in the overall performance of the hybrid model.

Operating Unit Data

Much of the data collected related to the organizational attributes, process performance, and financial metrics associated with the seven operating units that comprise TechSys. Detailed, verified data was collected for the division, the matrixed research and development unit, and one of the operating units. The data for other operating units are not as detailed, however. For instance, while budget numbers and total headcounts are well known for every operating unit, the exact division of labor between program management, engineering, and business development is not known for many of the other operating units. These numbers, however, have been estimated by individuals with experience at the multiple operating units, following a common heuristic for division of labor in given markets. These estimated parameters should be subjected to sensitivity analysis; if it were the case that the model was sensitive to subtle changes in one of the estimated parameters, further work would have been done to collect more accurate data.

Qualitative Variables

Some of the variables developed in the previous steps to describe the relationship between the operating units are highly qualitative, such as “process compatibility” or “strategic complementarity” between operating units. These variables are subjective, and best determined through survey. The values used in the model come from a survey taken by a subject matter expert at the TechSys division level. A Likert 5-value scale was used to classify the nature of each qualitative relationship. Table 6-5 lists these qualitative comparison values along with the valuation criteria used to classify each variable.

These qualitative values were incorporated into the model by creating functions that took the qualitative values as inputs and output quantitative value that could be used as a multiplicative factor elsewhere in the model. For instance, the qualitative variable “market attraction” was used to determine the frequency of synergy opportunities between operating units. A score of “1” indicated that there would never be a synergy opportunity between two OUs; a score of “5” corresponded to 40% of all opportunities for an OU would be synergy opportunities with another OU³⁶. These values were determined using an internal TechSys study performed to examine synergy opportunities involving newly acquired OUs as well as older OUs. The relationship between the qualitative “market attraction” variable and the number of annual synergy opportunities was extrapolated based on this single study. Obviously, there will be a great deal of uncertainty associated with these values, so the model should be tested for sensitivity to these qualitative variables during model testing.

³⁶ The value of 40% was developed by a subject matter expert, based upon the maximum likely amount of synergy in a given operating unit.

Table 6-5: Qualitative operating unit comparison variables and their valuation criteria

Qualitative Comparison Variable	Valuation Criteria
Market Attraction - Is there an attraction between the markets pursued by these two operating units that leads to joint development?	1. No attraction whatsoever between markets 2. A single example of collaboration exists 3. A few examples exist, but are considered isolated 4. There is occasional attraction between the markets pursued 5. There is frequent attraction between markets; these are natural collaborators.
Personal Relationships	1. No personal relationships other than dictated by management 2. Few personal relationships exist between operating units 3. Some horizontal relationships exist between operating units; 4. At least one person in each department knows someone else in the corresponding department. 5. There are many strong personal relationships between operating units; inter-OU communication is very common
Effectiveness of IT infrastructure for collaboration	1. Non-existent 2. Existent, but ineffective for joint operations 3. Somewhat common infrastructure. Enhances some collaboration between OUs 4. Common IT infrastructure; some semantic and tool mismatches 5. Seamless integration of all IT systems, tools, and semantics
Process Compatibility	1. Processes completely incompatible. 2. Processes highly different, but can be forced to work together 3. Processes different, but work together with some caveats 4. Processes similar, but designed for compatibility 5. Processes identical

6.8.2 Simulation Software

The TechSys hybrid simulation model was implemented using the AnyLogic software platform. AnyLogic is a flexible development platform for the development of deterministic or non-deterministic simulation models employing a wide array of simulation methodologies. This allows the creation of all three sub-

models within the same development environment, using the same timing and debugging engine, which greatly reduces the technical hurdles associated with creating hybrid simulation models. AnyLogic is a JAVA™- based application, and has the ability export the model into portable JAVA bytecode that can be run on any computer platform that has JAVA installed. Because of this, the model can be freely distributed without an AnyLogic software license.

Following the object-oriented programming paradigm, AnyLogic allows the creation of models with multiple levels of encapsulation and inheritance, which makes the operating unit-centric topology described in Section 6.7 possible. It also has tools for the development of graphical interfaces to the model that can be highly customized. Figure 6-10 shows the AnyLogic graphic model of a generic TechSys operating unit. Every rectangle in the sub-model view is an encapsulated model can be viewed by selecting it. Parameters can be adjusted in a single table, while the initial value of variables can be adjusted by selecting it on the model chart.

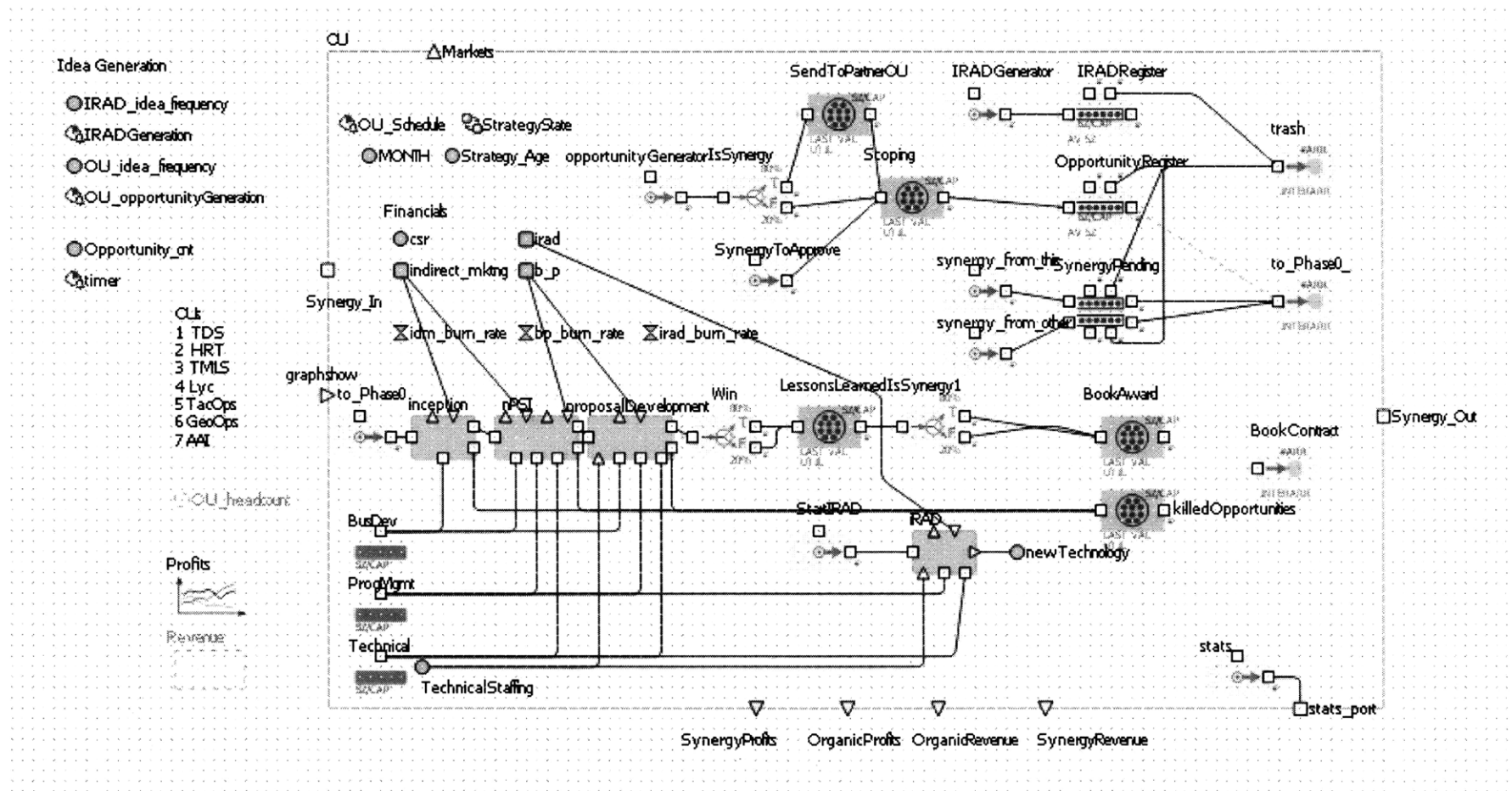


Figure 6-10: The operating unit sub-model in the AnyLogic development environment

The high-level of flexibility afforded by AnyLogic comes at a price, however. The learning curve to produce models of comparable complexity to that of the TechSys simulation model is quite substantial. Without previous experience with JAVA programming, the learning curve is even steeper. Despite these difficulties, it is difficult to recommend any other programs or approaches to creating hybrid models. "Middleware" approaches, either custom developed or using a platform such as ModelCenter, are much more labor intensive to develop, run more slowly, and can be very difficult to debug, based on initial testing for use in an enterprise architecture based hybrid simulation environment.

6.8.3 Implementing the Process Sub-Model

Of the three sub-models in the simulation model, the process sub-model required the most additional development work to take it from the initial diagrams into a functional simulation. The primary challenge was to determine the operational form of the simulation: the discrete event simulation would need to have the flow of some type of entity that could capture resources and modify the performance of the process based on its characteristics. In the case of processes associated with the development and capture of new business opportunities, it was a logical step to create an entity in the sub-model that would represent an individual opportunity.

TechSys' opportunity identification, selection and development processes across its operating units handle a wide array of business opportunities, ranging from small opportunities with potential revenue in the thousands of dollars to very large opportunities worth several hundred millions of dollars. The opportunities have different customers, serve different markets, and require different technical knowledge. To adequately evaluate the architecture of this process, the entities used in the simulation sub-model would have to be equally diverse. Each real-world business opportunity is unique; initial observations at TechSys has shown that the irregularity of both the timing of opportunities and the resources required

to take advantage of them is responsible for many process bottlenecks, and the simulation should capture this aspect of the process dynamics.

Opportunity Entities

The process sub-model employs a set of software entities in the simulation flow to represent individual business opportunities. These “opportunity entities” flow through the process and have actions taken on them at discrete process steps and at defined decision points. Each opportunity entity is defined using a set of attributes unique to that entity, such as its cost or the probability that it will pass its initial gate review. These attributes can be divided into two types: those defined at entity creation, and those defined during the execution of the process. See Table 6-6.

Table 6-6: Opportunity entity attributes

<i>Defined at Entity Creation</i>	<i>Defined in Run-time</i>
<ul style="list-style-type: none"> • Opportunity Level • Complexity • Synergy status • Strategic Impact • Market area • Risk • Staffing resources required 	<ul style="list-style-type: none"> • Probability of passing the each process gate • Cost Estimate • Revenue Estimate • Net Operating Profit • Budget spent at each gate

When an opportunity entity is created, it is assigned attributes that define it throughout the process. These attributes include size, complexity, strategic impact, and risk. Each opportunity at TechSys is assigned a level designation (defined as “A”, “B” and “C”), which roughly correlates to values of risk, size, costs, revenues, and strategic impact. For purposes of generating opportunity

entities for the process sub-model, however, this process has been reversed; opportunity entities are created with an assigned level based on the historical distribution of opportunities pursued by a particular operating unit, and then the attributes associated with that level are assigned based on the distribution of values for that attribute and level. For example, at one operating unit, 66% of opportunities were “C”s, 22% “B”s, and 12% “A”s. Figure 6-11 shows the historical distribution of cost at one TechSys operating unit for Level C opportunities (those with low risk, mature technologies, and relatively low investment required).

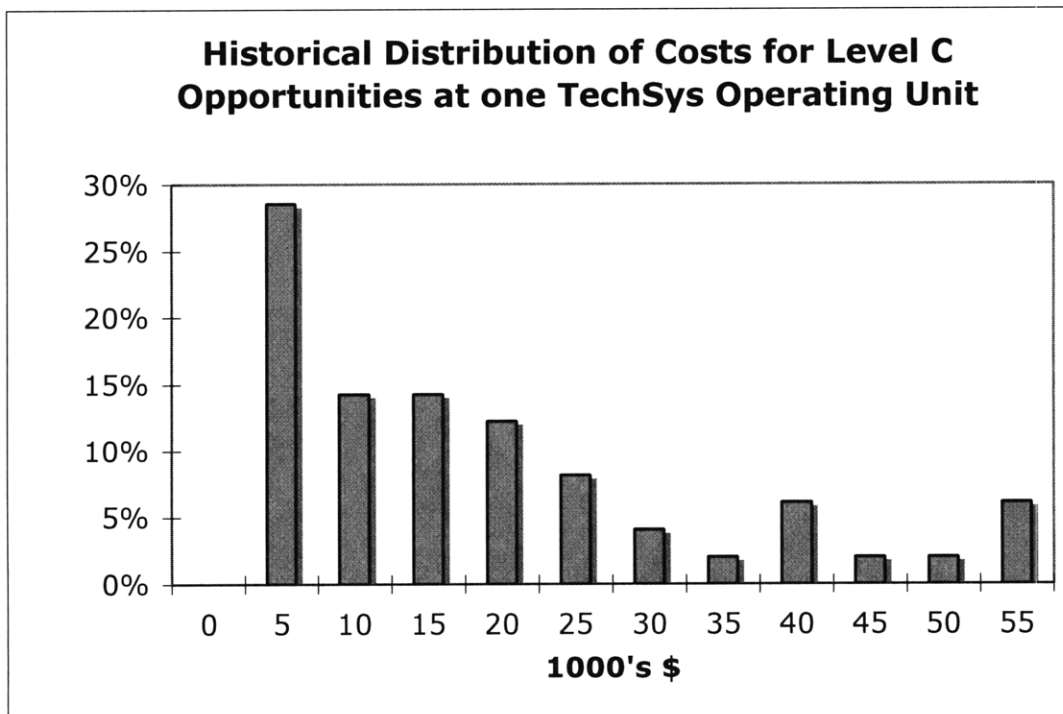


Figure 6-11: Historical cost distribution for Level C opportunities at one TechSys operating unit.

In addition to attributes that are defined at entity creation, there is another class of entity attributes that are defined during process run-time. These run-time attributes are temporary containers for values calculated during process execution, such as the calculated probability that an entity will pass a given gate

review. These values are assigned to the opportunity based on the execution of the process and will be used as input back into the process at a later step.

Process Timing

Each opportunity entity is probabilistic with respect to defined entity attributes as well as timing. Opportunity entities are continuously generated at uniformly distributed times, and with a frequency determined by the population of employees in business development. Just as in the real process, all potential opportunities are then held in a register, where they are reviewed twice a year, and a subset of all possible entities are selected to receive funding and undergo further development. Those opportunity entities selected to go through the gated review process spend a probabilistically determined length of time at each process step. This processing delay is chosen from a normal distribution with a mean chosen according to the opportunities size, and a variance associated with its complexity. The processing time is important, because while the opportunity is being processed, resources are consumed (the budget is spent) and resources are being tied up (program management staff, business development staff, and technical staff). If the budget has been exhausted or if no free resources exist, then no new opportunities can be pursued.

pursued.

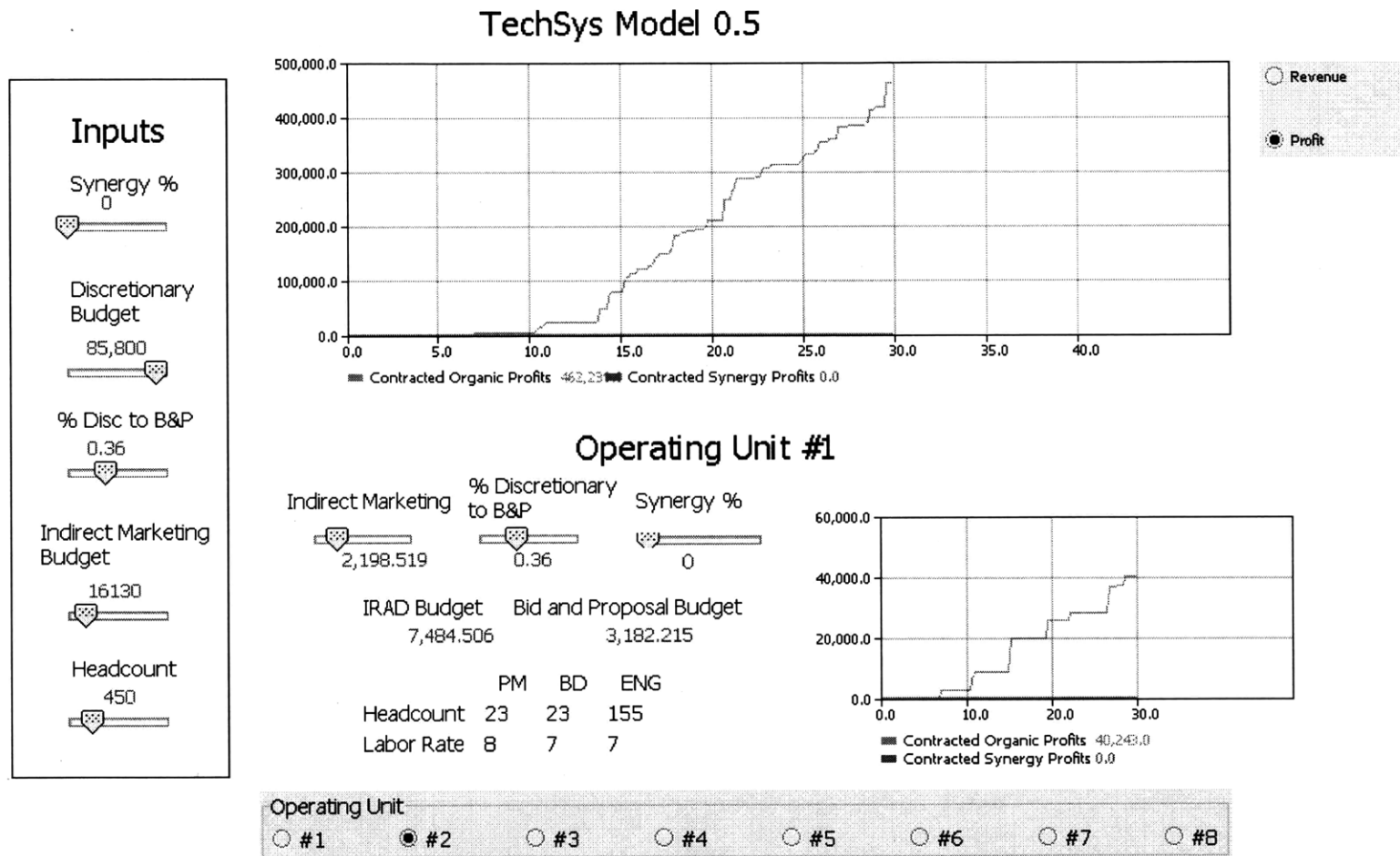


Figure 6-12: The TechSys Enterprise Architecture Simulation Model Interface (single replication)

6.8.4 Developing the User Interface

One element of model implementation that had not been addressed by previous steps in the modeling process is the development of a user interface for the model. The purpose of the user interface is to give the user access to the input parameters of the model, execute the model, and then observe the output. After receiving feedback from TechSys stakeholders, an additional interface was developed to modify the inputs and view the outputs of a single operating unit. Figure 6-12 shows the user interface of the model.

When the model is run for a single iteration, the cumulative profits or revenue can be graphed in the central chart area as the model executes. The model execution can be paused at any point, and the input parameters can be changed, allowing for a shift in resource allocations in the third year, for example. When the model is run in Monte Carlo mode, the graph will display the distribution of the cumulative value of the model output over the model's time horizon.

While the user interface allows the user to change input parameters, it does not provide a simple means to allow the user to change the architecture itself. In order to test candidate architectures against one another, the actual architecture of the model must be changed. This would entail changing some aspect of the model's structure, e.g. an incentive or rule. In such a case, two different models representing two different enterprise architectures would both be run, and their output probability distributions would be compared. For purposes of presentation, however, a "switch" element in the user interface could be used to toggle between the execution of two pre-defined enterprise architectures, but this would not allow the flexibility to run experiments with the architecture.

6.8.5 Running the TechSys Simulation Model

The TechSys enterprise architecture simulation model is intended to be used as a tool to explore the potential performance of an enterprise architecture over a range of conditions. Each time the model is executed, the outcome will be unique, as the process sub-model contains a number of non-deterministic steps. The key output is not the results of a single execution of the simulation, but rather the distribution of results from a number of executions of the simulation.

In keeping with the nomenclature of simulation modeling, a single execution of the simulation model using one set of input parameters is called a *replication*. Each replication will produce a unique result in a Monte Carlo fashion. A set of replications sharing the same input parameters is called a model *run*. Figure 6-13 is the performance distribution for a single run of the simulation with input parameters identical to those used in TechSys in 2007.

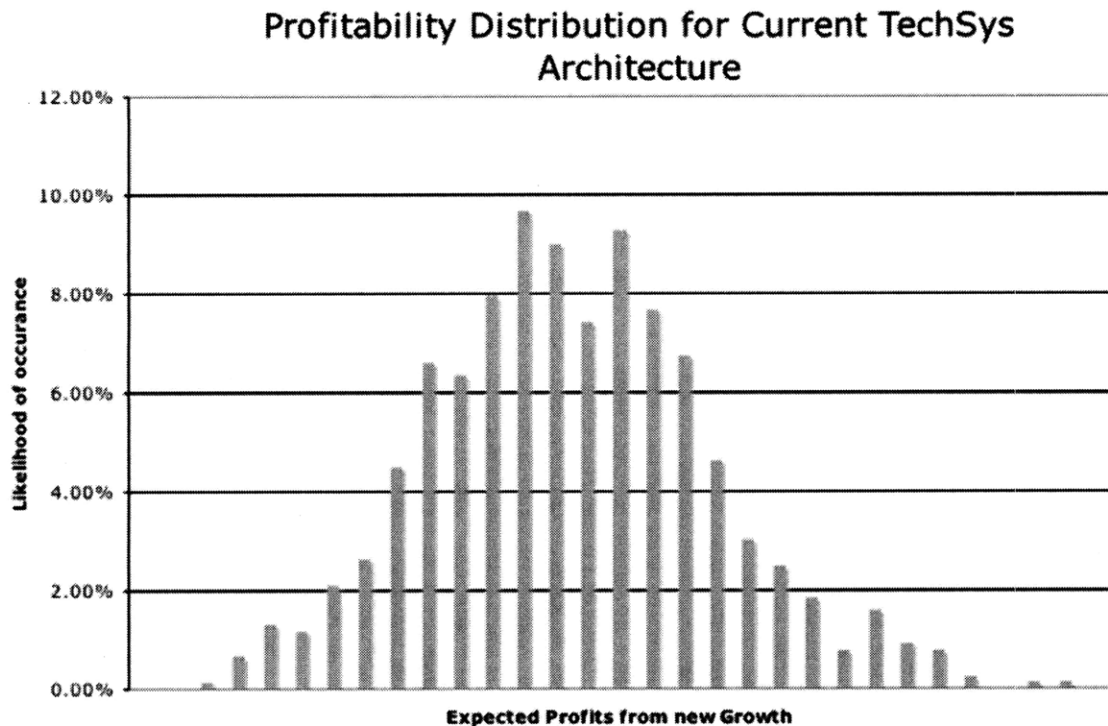


Figure 6-13: The histogram representing the combined profitability from organic and synergistic profits for a single simulation run of the current-state architecture. The independent axis values are withheld. N replications = 750

When comparing two different model configurations (either different inputs or different architectures), the probability distributions of their outputs should be compared, rather than the results of a single replication. Both the mean and variance of the output distributions should be compared. One model configuration, for example, might have a slightly higher mean than a second configuration but it may also have a higher variance. In this situation, the second configuration might be preferable since its lower variance would indicate lower risk and greater predictability.

In Monte Carlo analysis, the modeler must choose the number of model replications to be executed for each run of the simulation. If this number, N, is

too low, the output distribution will be insufficiently sampled, and any measure of the mean or standard deviation will have a low degree of confidence. If N is too high, computation time will be wasted. As a point of reference, each simulation replication, using a time horizon of 3 years, take approximately 20 seconds to run in AnyLogic 5.5 on a machine with a 2 Ghz Intel Core Duo 2 processor with 2 GB of memory. This execution time means that simulation runs with replications numbering in the thousands is highly undesirable. It would be advantageous to use the smallest N possible while having confidence in the model's results.

One approach to determining the value of N is to continue to run the simulation until the moving mean and moving standard deviation³⁷ of the replication outputs becomes stable. This condition indicates that there are enough replications that any statistics performed on the run are valid.

Figures 6-14 and 6-15 show the moving mean and moving standard deviation of the simulation run as the number of replications in the run increases. The dependant axis in these figures have been normalized by showing their value after N replications as a percentage of the running mean or running standard deviation at the end of the run.

³⁷ The terms "moving mean" and "moving standard deviation" are used to indicate the values of the mean and standard deviation of the output distribution recalculated after each new replication.

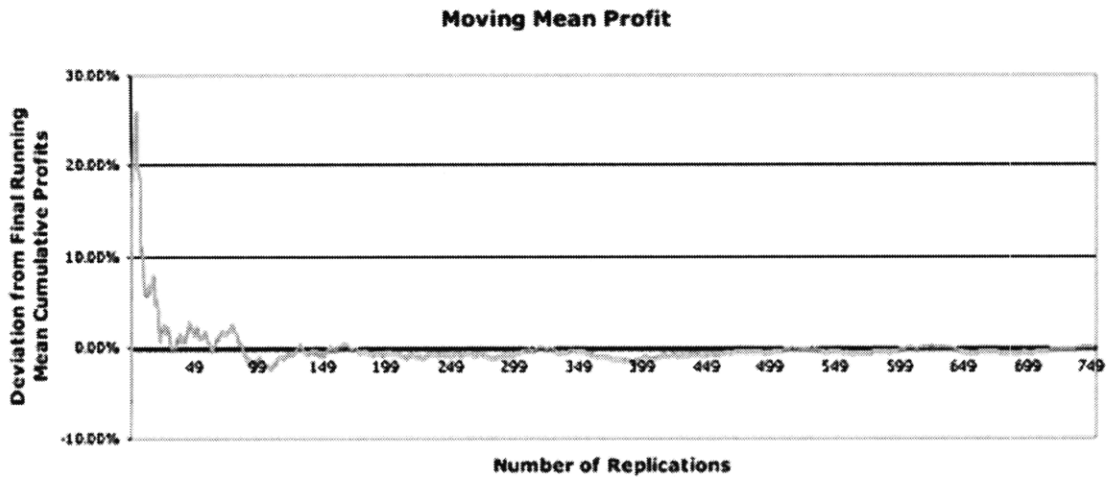


Figure 6-14: A graph of the moving mean of the cumulative profit from the TechSys model, using 2007 input parameters and the current architecture.

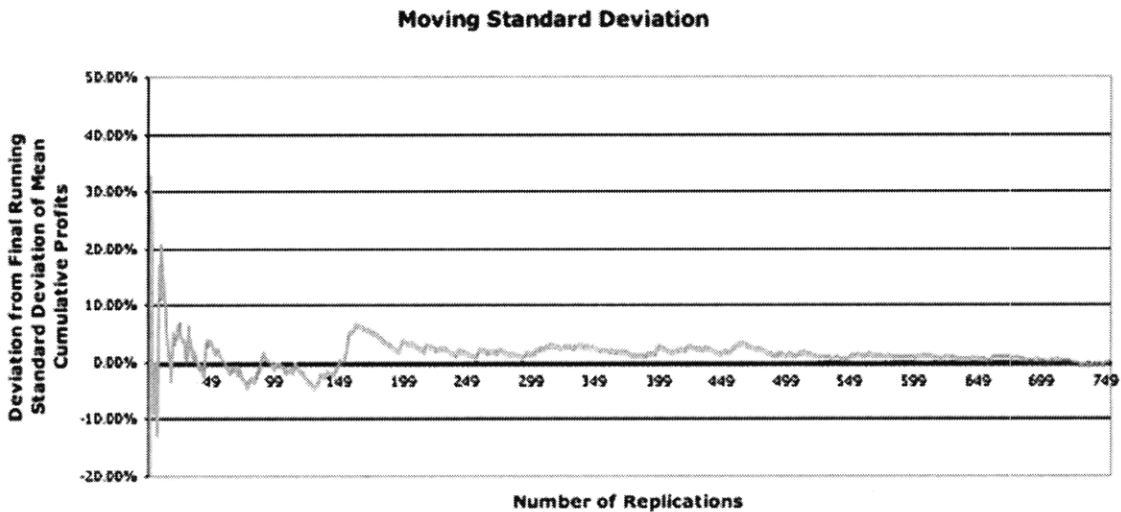


Figure 6-15: A graph of the moving standard deviation of the cumulative profit from the TechSys model, using 2007 input parameters and the current architecture

By inspection of Figures 6-14 and 6-15, it can be observed that as the number of replications increases, both the mean and standard deviation converges on a steady state value. After approximately 200 simulation replications, both the mean and standard deviation have reached relatively stable values within 1% of final values after 800 replications. This analysis was repeated for different

model parameters with similar results: the model tends to produce a very stable distribution after 200 replications.

As a result of this analysis of the simulation's output variability, 200 model replications were used in each simulation run to determine the mean of the model's output for a given set of input parameters. Any chart or graph of simulation model performance shown in this chapter uses the mean of a performance distribution as its data.

Next Steps

After the hybrid model has been implemented, must be tested in order to build confidence in its design, operation, and results. Chapter 8 will discuss how the model was tested and analyzed to build confidence in the model's operation. This analysis is another place in the process where iteration is likely to occur: as problems are uncovered, the modeler may return the model, its abstractions, structure, and data and make updates. After the model testing, Chapter 8 will discuss the findings made using the TechSys hybrid simulation model.

Chapter 7: TESTING AND ANALYSIS OF THE TECHSYS ENTERPRISE ARCHITECTURE SIMULATION MODEL

“All models are wrong. Some are useful”

- George Box(1979)

The TechSys Enterprise Architecture Simulation model is only useful to TechSys if its results can be trusted enough to influence decisions regarding future changes to the enterprise architecture or to the way that the enterprise is managed. This chapter details how the model was tested with the participation of TechSys stakeholders in order to build confidence in its performance and understand its capabilities. With this confidence established, the model is then used to answer questions of critical importance to the strategic direction of the enterprise. After using the model of the current state architecture to gain an understanding of the structures driving undesirable enterprise dynamics, an alternative architecture is developed and analyzed to address the shortcomings of the current state architecture. The chapter concludes with a discussion of the benefits of the hybrid, enterprise architecture based approach for TechSys and potential future uses of the TechSys simulation model.

7.1 TESTING THE TECHSYS SIMULATION MODEL

After successfully integrating the sub-models of the TechSys enterprise architecture simulation model and running the resulting hybrid simulation, the next task was to test the model in order to build confidence in its performance. The course of model testing was designed to uncover errors in the model's formulation and calibration, expose the model's limitations, and improve its ability to provide insight into the architecture of TechSys with enough confidence that it can be used as an input into the decision making process. Additional tests were conducted to ensure that the model can reasonably account for the historical performance of the enterprise, while placing the majority of emphasis on understanding the structure of the architecture rather than on any predictive capabilities.

A enterprise architecture simulation model does not provide point predictions, but rather indicates the range of behaviors that are possible and likely given specific structures and policies that comprise its architecture. The performance of the enterprise is dependent on the specific contracts that are won in a given year, specific externalities such as the economy, and tactical and operational performance, none of which this model of the enterprise architecture captures. Rather than focusing on attempting to predict such events, the simulation model of the enterprise architecture helps to identify the strengths and weaknesses of a given architecture over a wide range of probabilistic events. For this reason, it is difficult to think about testing the model using a traditional "verification and validation" approach.

In keeping with the guidance of Sterman (2000) and Forrester (1962), the terms "verification" and "validation" are not used in testing this model because they imply a certainty or accuracy that simply is not present in high-level models of

truly complex systems. This does not imply that the model should not be rigorously tested; it simply means that the model should be evaluated with an eye towards its intended purpose of enterprise architecture evaluation rather than point predictions of enterprise performance.

The following sections describe the methods used to test the TechSys simulation model, as outlined in Section 4.6.9.

7.1.1 Testing for Boundary Adequacy and a Structural Assessment

The first tests of the model occurred during model creation as part of a continuing dialog with TechSys stakeholders to identify proper model boundaries. The first question asked of the stakeholders was “what are the important concepts and behaviors that the model must address endogenously?” Early stakeholder interviews were used to determine the boundaries of the enterprise architecture that affected the dynamics of growth. One of the most significant changes to the early conception of the model was to expand the model boundaries to include internal research and development (IRAD) activities within the model scope. Upon first glance, IRAD appeared to be a non-essential component of the dynamics of cooperation between operating units. A re-examination of model boundaries showed that IRAD investment was to be a fairly critical dynamic behavior that did affect cooperation and exploration of new business between operating units. Without specifically including IRAD as endogenous to the model, an entire feedback loop would be missing from the model, significantly undermining its ability to address the enterprise’s ability to make investment tradeoffs between exploitation and exploration of its environment.

A second boundary that was reconsidered during model testing was the assumption of a three-year time horizon for the model. Although all stakeholders agreed that a three-year time horizon was appropriate given TechSys’s planning

cycles, the time horizon was changed to both a five and ten years to determine the difference this made in model performance. A further analysis of this boundary test is discussed in Section 7.3.2

As the model was constructed, stakeholders reviewed basic assumptions of the model. First, the structure of the model (processes, incentives for the operating units) was reviewed to ensure that it was consistent with everyone's understanding of the enterprise architecture. This was done at a series of model review meetings, where the stakeholders were walked through the assumptions and logic of the model step at a time. Where discrepancies were noted between the model and common understanding of the architecture, changes were made to the model. The model served as a communication tool in these review meetings, helping everyone in the room to see the enterprise architecture from the same perspective, which previously had been difficult to do.

In addition to basic structure, decision rules within the model were reviewed, including how operating units decided which proposals and IRAD projects to pursue and how resources were allocated to pursue them. For example, upon review, it was discovered that the model did not allocate engineers between proposal development and IRAD activities using realistic decision rules. This allocation was revisited and new rules were developed that more closely modeled the resource allocation in practice.

7.1.2 Parameter and Variable Assessment

While performing the boundary and structural reviews of the simulation model with stakeholders, input parameter values used in the model were also reviewed. In many cases, several low-level model parameters such as engineering labor rates, headcounts across operating units, and budgets for the most recent two

fiscal years could be verified with documentation that TechSys provided. In other cases, input parameters had to be extrapolated from the other data sources and used to create probability distributions, such as the distributions for the number and size of proposals and IRAD projects. A few of these quantitative parameters, such as the probability of a proposal passing each gate review, the statistics on processes timing or the maximum number of engineers allowed to work on IRAD at any given time, could not be supported by recorded metrics or processes at TechSys³⁸. In these cases, values were selected based on the opinions of subject matter experts. The qualitative parameters used in the model to describe “soft” attributes between operating units, described in Section 6.8.1, were based on stakeholder surveys.

After checking the inputs into the model for accuracy, the model’s output variables were analyzed to ensure that model execution was consistent with historical data. Key intermediate model outputs, including the annual number of completed IRADs or the average size of successful proposals, were checked against available records to ensure that the past performance of the enterprise was within the output range of the enterprise architecture simulation model. No statistical tests were performed to test the correlation of the model output with the historical performance because the model is not deterministic—any one replication of the model is randomly generated, and the output of a full model run of 200 replications indicates an average performance given a wide range of inputs and conditions, rather than a replication of past performance.

For testing purposes, the simulation model was run with input resource parameters (budgets, resources devoted to synergy, headcount) identical to those in use in 2007, and in each case the historical data points within the distribution produced by the simulation model. Figure 7-1 shows the output distribution for the annual count of successful proposals produced by a single run

³⁸ In the case of process timing, process documentation did give guidelines, but interviews suggested that this guidance was optimistic.

of the model (200 replications). The dark vertical lines indicate the two historical reference points for which data is available. These lines fall within the distribution produced by the simulation model, which should be expected.

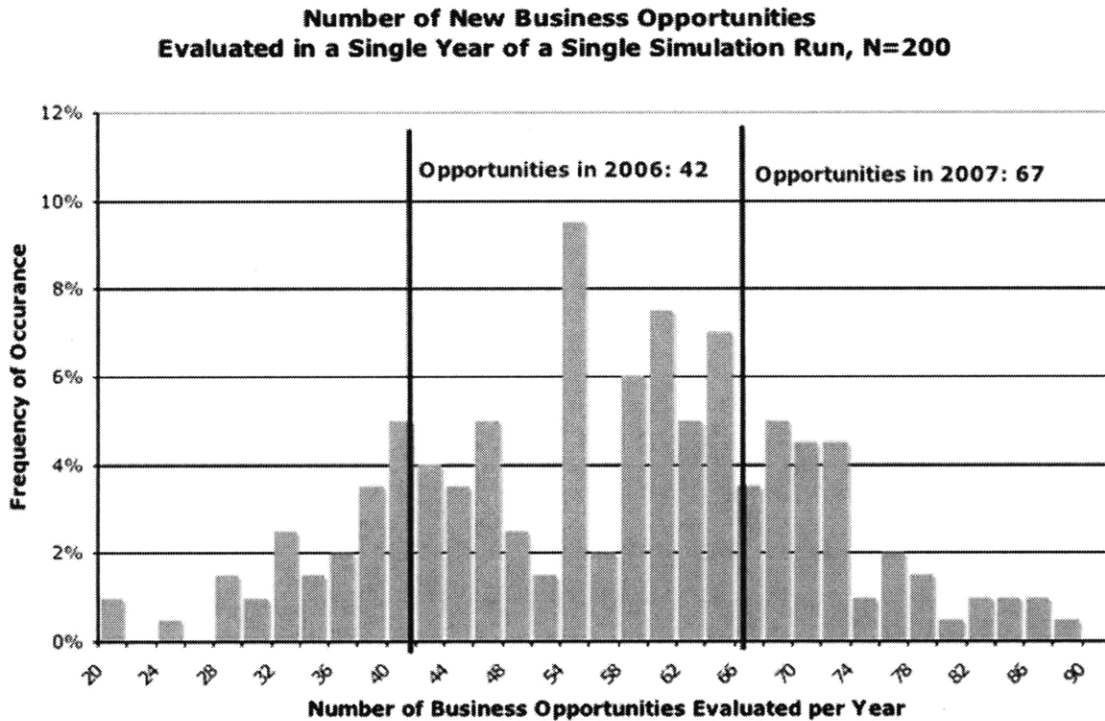


Figure 7-1: Distribution of the number of business opportunities evaluated in one year of the simulation for OU #1 for a single simulation run with 200 replications.

Figure 7-2 is another test of model performance that compares the size of business opportunities as produced by the model against their historical size, using the amount of Bid and Proposal money spent on each opportunity as a proxy for size. There are three different “levels” of business opportunities: Level C, which very roughly represents smaller opportunities (under about a \$60,000 investment), Level B (mid-sized) and Level A (very large, risky, or strategically important opportunities over \$100,000). Each level has its own distribution of costs, so each level was evaluated independently. Figure 7-2 shows the distribution of sizes of Level C opportunities in Operating Unit #1, which roughly follows an exponential distribution. The blue bars show the historical distribution,

while the red bars show the distribution produced by a single run (200 replications) of the simulation model. As can be seen, the model comes very close to replicating the historical distribution, with an $R^2 = 0.87$, indicating a high correlation. This should be expected, because the distribution that the model uses to create new opportunities is based on this historical data.

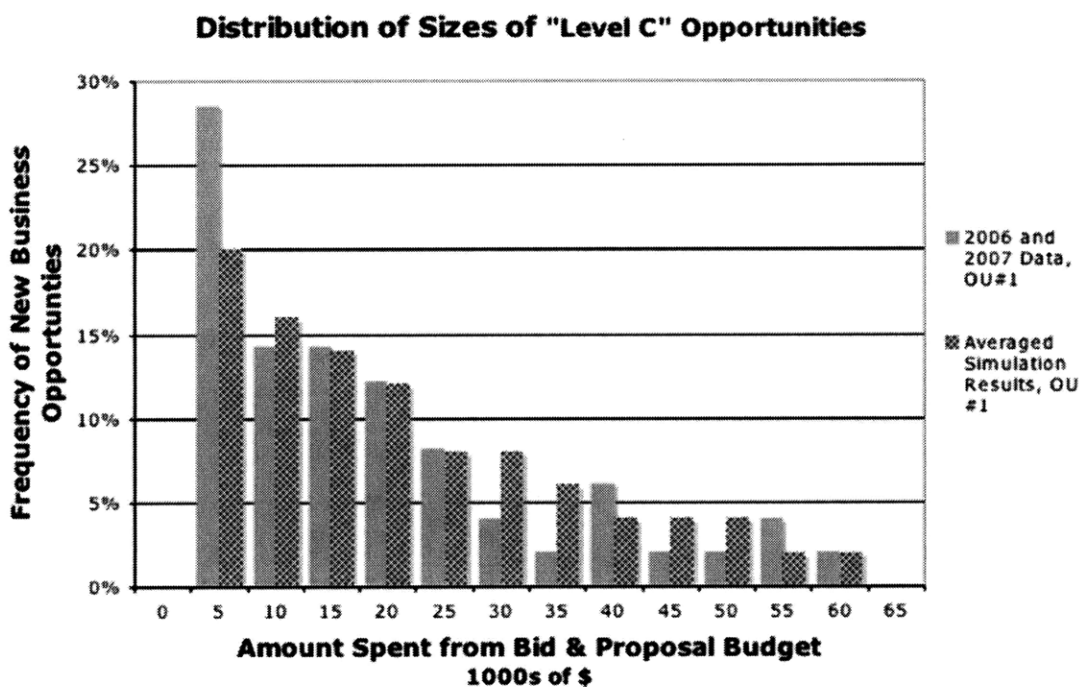


Figure 7-2: Distribution of opportunity sizes produced by the model and from historical data. $R^2=.87$, Mean Absolute Error = 2%

7.1.3 Extreme Condition Testing

Extreme condition testing of simulation models tests the formulation and robustness of the model by examining the model’s behavior when inputs are taken to the extremes. This serves as a check to ensure that the model can be trusted to behave in a realistic fashion regardless of extreme inputs or conditions

in the model. Often such tests can serve to uncover unseen flaws in model formulation that do not appear during “normal” operation of the model that will certainly come into play while testing the model’s performance over a wide range of inputs and conditions. Within the TechSys model, examples of extreme condition testing included:

- If there is no money left in the IRAD budget, are there new IRAD starts?
- If there is no money allocated to IRAD, what happens to long-term profitability?
- If the entire Discretionary budget is devoted to IRAD and none is given to Bid & Proposal, does TechSys win any contracts?
- If all engineers are tasked with working on a large proposal and more are required, would a new IRAD receive engineers first?
- What happens as the time horizon of the model is extended?

Much of the extreme condition testing was performed during model creation, and helped to resolve many problems surrounding resource allocation. All variables were checked for unrealistic behavior (such as a negative budget balance, which was discovered during model testing) and resolved when discrepancies were noted.

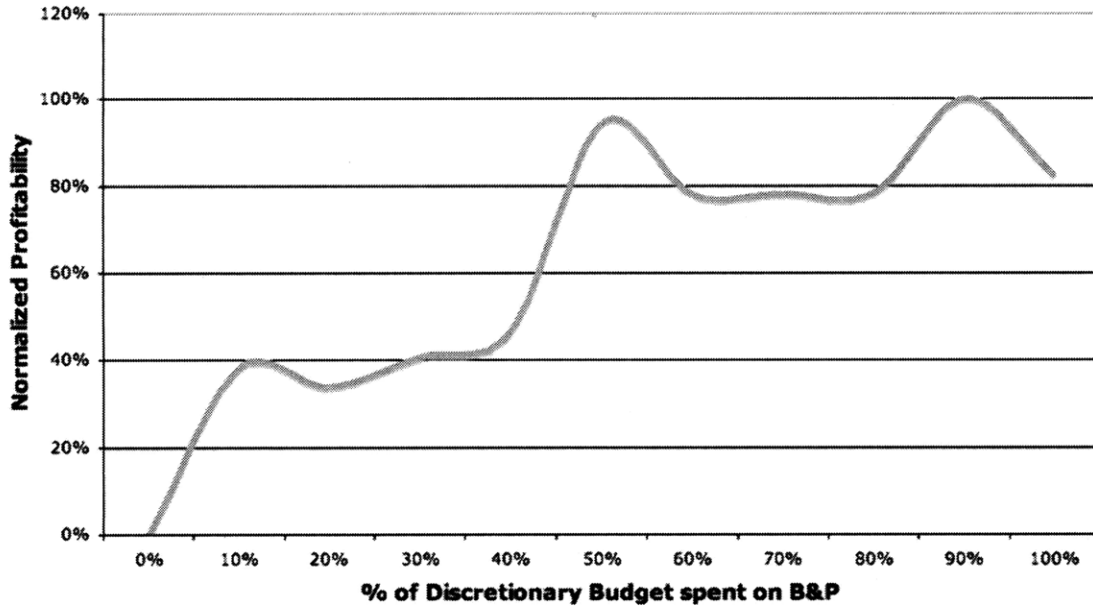


Figure 7-3: Normalized Profitability of TechSys as a function of allocation of the Discretionary budget to Bid and Proposal, with the remainder allocated to IRAD.

Extreme condition testing uncovered an important assumption made in the model. When no budget is allocated for Bid and Proposal, TechSys does not win any new contracts, as expected. When the budget is entirely allocated to Bid and Proposal and none to IRAD, however, profitability decreases, but not substantially. See Figure 7-3 for a sensitivity analysis of expected profitability as the allocation of the Discretionary budget is varied.

It was assumed that the long-term profitability of any high-tech firm would be dependant on investing in research and development. If no resources were allocated to research and development, profitability should eventually suffer. Why is profitability not impacted by lack of IRAD investment in the model? The initial observation was that the time horizon of the model (three years) was simply too short to show the impact of under-funding research and development. This made sense to TechSys stakeholders, because the return on investment of any research and development activity was seen to be greater than three years. However, upon running the model using five and ten year time horizons, an

expected difference was not observed. After reexamining the model, it was discovered that the model did not distribute the value of research and development over time, but rather allocated its benefits upon completion of the IRAD project. This indicated that the model as formulated at the time was incapable of capturing the detriment due to underfunding research and development, and it was not attributable solely to the time horizon used by the model. As a result, changes were made to the model to capture these effects.

7.1.4 Sensitivity Analysis

One of the most valuable characteristics of the TechSys enterprise architecture simulation model is the ability to test the sensitivity of the enterprise architecture to variations key parameters or variables. Sensitivity analysis can be used to evaluate the model formulation for artificial sensitivities to parameters that are not present in the real-world system, indicated a flaw in the model formulation. It is also very useful for exploring the tradespace of the model, exploring the effects of different combinations of inputs and strategies. Because the TechSys simulation model relied heavily on sensitivity analysis in this latter use, the full discussion of sensitivity analysis testing is reserved for Section 7.2, Analysis using the TechSys Simulation Model.

7.1.5 Summary of Model Testing

The TechSys simulation model was subjected to a wide array of tests designed to explore its capabilities, build confidence in its performance, and identify its weaknesses. Model testing was performed with feedback from TechSys stakeholders, who were a central part of testing and provided essential feedback. Their involvement not only improved the model, but also increased their trust in the model allowing it to be more effectively used as a tool for analysis of the TechSys enterprise architecture.

7.2 ANALYSIS USING THE TECHSYS SIMULATION MODEL

At the highest levels of control, TechSys had a limited number of levers at its disposal to influence enterprise growth:

- The percentage of resources devoted to pursuing synergistic business opportunities;
- The Discretionary budget;
- The amount of the Discretionary budget allocated to Bid& Proposal versus Internal Research and Development;
- Headcount;
- Indirect Marketing Budget;
- Changing the enterprise architecture itself.

For purposes of analysis using the simulation model, TechSys desired to keep the headcount and Discretionary budget input parameters fixed, implying that no new resources would be used to foster new growth in the analysis. This left three options for influencing new enterprise growth in the model: the percentage of all new business opportunities that are synergistic, the percentage of the Discretionary budget allocated to Bid and Proposal (with the remainder going to Internal Research and Development), and changing the architecture itself. The first step in the analysis was to test the current architecture “as is” to develop a baseline for any changes.

Assuming that the current enterprise architecture is fixed, there were two key parameters that could be varied: the percentage of new business opportunities that are synergistic, and the percentage of the Discretionary budget that is allocated to Bid and Proposal. In the first analysis, the allocation of the Discretionary budget was held constant, and only the percentage of new business opportunities that are synergistic were varied.

7.2.1 Investment in Pursuing Synergy

The expectation of the existing TechSys strategy is that as the percentage of synergistic business opportunities is increased relative to the amount of organic opportunities, the overall profitability of TechSys should also increase³⁹. The reasoning for this is that synergy opportunities build the foundation for growth into new, more profitable markets. In theory, by pursuing synergy between its operating units, TechSys should have a stronger, more competitive position in the market. Over the preceding four years, TechSys's synergy-driven strategy influenced many of the decisions that led to the development of the current state TechSys enterprise architecture. As can be seen from the sensitivity analysis of Figure 7-4, however, the output of the TechSys enterprise architecture simulation model tells a dramatically different story with regards to the benefits of pursuing synergy with the current state TechSys enterprise architecture.

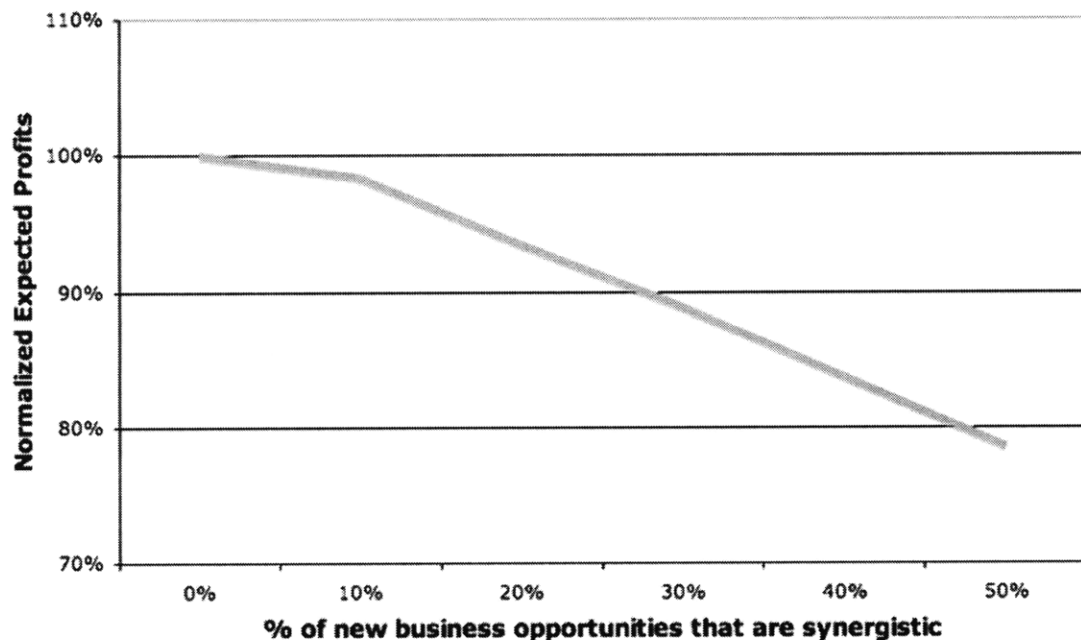


Figure 7-4: Expected profit as synergy investment is varied

³⁹ For a review of TechSys's usage of the terms "synergy" and "organic growth," see Section 6.2.1.

Figure 7-4 indicates that in the TechSys enterprise architecture simulation, *as the percentage of new business opportunities with a synergy component increases, enterprise profitability will decrease*. This result strongly contradicts the prevailing theory of the effectiveness of pursuing synergy. On the surface, diminishing returns from pursuing a strategy based on synergy seems illogical. Upon examination of the behavior of the model, however, the reason for this significant inconsistency emerges directly from the enterprise architecture itself rather than from any shortcomings of effectiveness of pursuing synergy as an idea. The crux of the issue is that both the process and organizational architectures are structured such that synergy opportunities are systematically not selected compared to organic opportunities, leading to a large opportunity cost when more effective organic opportunities could have been pursued.

Perhaps one of the reasons that this shortcoming of the architecture had not drawn more attention previously is that it lies at the intersection of multiple views in the enterprise architecture framework. There are two key contributing factors that cause this process to favor organic opportunities: the first is the structure of the selection process itself, and the second is the organizational incentives of the operating units. These factors when combined have a multiplicative effect, causing the process's selection bias to be worse than analysis of the process and organizational incentives independently would suggest. The following sections will present the problem from the perspective of each sub-model, and then show how they interact to compound the problem.

Process Sub-model Perspective

In the model, all synergy opportunities are *independently* evaluated by each participating OU and prioritized against all other opportunities that each OU has. There is no central TechSys level overview of synergy opportunities. For a synergy opportunity to be funded for further consideration, it must be independently chosen by all participating OUs during each OU's biannual

Opportunity Review, where OUs pick which new business opportunities to fund in the coming months using funds from their Bid and Proposal and Indirect Marketing Budgets. Because synergy opportunities must be independently approved twice by OUs with different local incentives, synergy opportunities are not selected at a rate exceeding new “organic” opportunities.

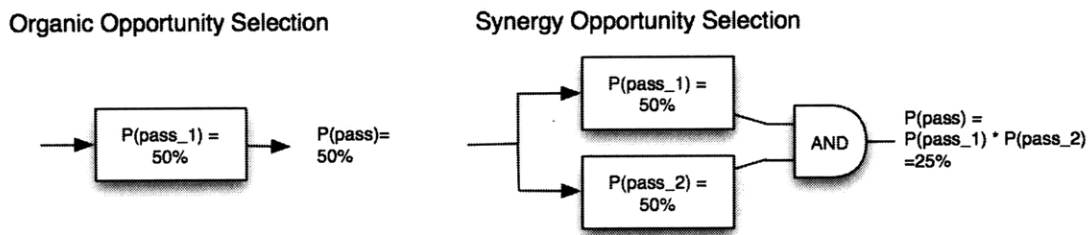


Figure 7-5: The mechanics of Synergy and Organic Opportunity Selection

Figure 7-5 illustrates the mathematics of the opportunity approval process for both organic and synergy opportunities. On the left, an organic opportunity must only pass a single review. In the case of a synergy opportunity, it must pass two independent reviews. This has the effect of a Boolean logic “and” gate. The resulting probability of the synergy opportunity receiving funding is equal to the product of pass rates from each OU’s review, which can substantially lower the overall pass rate.

Because synergy opportunities are more likely to fail early in their development due to the selection problem above, devoting resources towards developing new synergy opportunities limits resources for developing good organic opportunities that are more likely to pass. This results in fewer high quality (in terms of profitability and competitive positioning) organic opportunities available for selection. With a smaller selection of organic opportunities to choose from, the expected value of opportunities that receive funding decreases.

The Organizational Perspective

Taking an organization perspective, the problem can be viewed in terms of local incentives for the operating units. The opportunity review selection team funds those business opportunities that are best poised to further the competitive position of the OU, based on a return to the OU of both profit and competitive position. On synergy opportunities, the profits from an opportunity are not evenly split between OU partners, nor is competitive positioning. Typically, there is a synergy partner that accrues the majority of benefit from pursuing the opportunity. The OU with the smaller share of profits and competitive positioning often finds it more advantageous to pursue its own locally developed organic opportunities, causing the synergy opportunity to go unfunded by both OUs.

The example in Figure 7-5 shows that each review has the same probability of passing an opportunity; in this example, $P(\text{pass}) = 50\%$. This is not a realistic assumption, however. In practice, one OU will typically benefit more (either in terms of profit or competitive positioning) from a synergy opportunity than its partner will benefit, leading the total probability of a synergy opportunity passing to be *lower than the OU with a lowest probability of passing the opportunity*. In an example, if OU #1 has a 75% chance of passing a synergy opportunity and OU#2 has only a 25% chance of passing the same opportunity, the opportunity will have only an 18.75% chance of being funded.

This organizational behavior seems to occur despite the fact that the incentives of every General Manager are aligned to promote the profitability of the corporation (BigTechs) and the division (TechSys) before that of the local OU. Interviews and discussions with TechSys stakeholders revealed two reasons why OUs still may not act in accordance with these incentives. First, the GMs have profit and loss responsibility for their OU. Despite their financial incentive plan, they tend to focus on the part of their compensation equation that they have the most impact on: their own OU. In TechSys's history, it has been exceptionally rare that OUs have worked together towards a common goal, providing little

experience on whether and to what extent the GMs of the local OUs are willing to make decisions that might be detrimental at the business unit level while their actions help achieve global company-wide goals. The second reason given that might help to explain the lack of effective incentives is that while the GM and some of the higher level director positions may have a corporate and division level financial incentive scheme, the majority of people working on developing and selecting new business opportunities do not. Local incentives plus a lack of experience in working in a collaborative fashion lead to locally sub-optimized behavior.

Additional Cultural and Communication Barriers to Synergy

Additionally, synergy opportunities are more likely than organic opportunities to fail product lifecycle review gates even after they pass the initial Opportunity Review in the development pipeline. Communication difficulties and cultural alignment issues between operating units, included in the simulation model, cause the average failure rate for synergy opportunities to be higher than the average failure rate for organic opportunities, although the gap narrows over time as communication and cultural barriers are lowered due to repeated interactions between operating units (this forms an initial barrier to cooperation that will lessen over time).

The organizational sub-model incentives compound the bias found in the process sub-model for selecting synergy opportunities. While the parallel selection requirement of the process significantly lowers the likelihood of a synergy opportunity being selected, the problem is exacerbated when there is a large differential in benefit and preference among the OUs, as is often the case in practice. Even if an individual synergy opportunity was vital to the strategy of one operating unit, it could be rejected because it would not benefit cooperating OUs as much as other local organic opportunities, despite the fact that TechSys as a whole would benefit more from the synergistic project. Given TechSys's

The TechSys Enterprise Architecture Simulation Model was used to analyze the effect of varying the allocation of the discretionary budget under the current state enterprise architecture. All model input parameters, other than “Discretionary Budget allocated to Bid and Proposal,” were held constant at their 2007 levels. The discretionary budget allocation to bid and proposal was varied from 0% to 100%, corresponding to the extreme cases when all discretionary money would be allocated to either IRAD or Bid and Proposal.

The result of varying the discretionary budget allocation in the simulation model is shown in Figure 7-6. The graph shows the expected profits over the three-year time horizon, normalized such that 100% is equal to the maximum value possible.

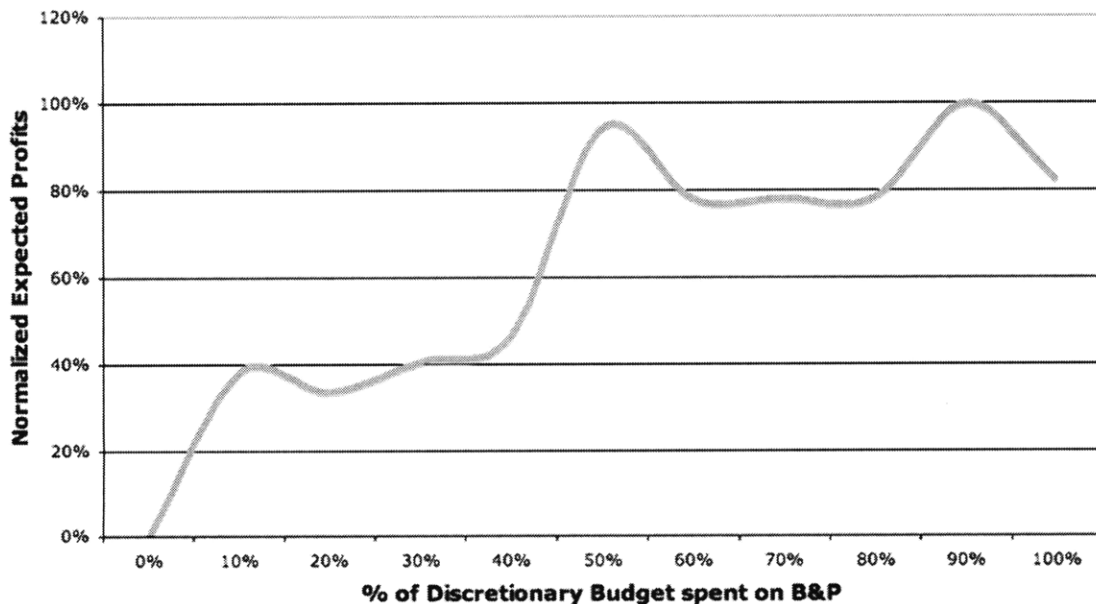


Figure 7-6: Normalized Expected Profit as the allocation of the Discretionary Budget is varied between IRAD (0%) and Bid and Proposal (100%)

As can be seen in Figure 7-6, the simulation model indicates that given the current architecture, the preferred investment strategy is to allocate the majority

strategic direction, this problem should be considered a critical area for improvement.

TechSys Corroboration

These results and the mechanism were corroborated with TechSys stakeholders. This behavior has been observed in practice, but synergistic business opportunities have remained such a small part of the total number of business opportunities considered to date that the systemic nature of the problem had not been highlighted. A working group at TechSys had identified the synergy opportunity selection process and incentives for OUs to pursue synergy opportunities as barriers, but had not realized the extent to which this barrier affects the system. This corroboration helps to confirm that this behavior is not purely an artifact of the modeling process, but is in fact present in the enterprise.

7.2.2 Allocating the Discretionary Budget

After understanding the effect of investing in pursuing synergy opportunities, the second major lever into their enterprise architecture that TechSys wished to investigate was the effect of the tradeoff made when allocating the Discretionary budget between the Internal Research and Development budget (IRAD), and the Bid and Proposal budget. As mentioned in Section 6.2.2, the discretionary budget is used defense contractors to fund both internal research and development, as well as bid and proposal activities. The discretionary budget must be divided between these two activities, creating a practical manifestation of the “exploration versus exploration” tradeoff common in the contingency theory literature. For many years, TechSys had allocated its discretionary budget without an explicit, quantitative analysis of the impact of this allocation on their profitability. TechSys management clearly understood that investment in IRAD was necessary to ensure future growth, but also knew that without investing in pursuing new business using their existing capabilities, they would receive no new business. The simulation model can be used to better understand how this allocation decisions impacts the performance of their enterprise.

of the Discretionary budget to the Bid and Proposal budget, and less to the IRAD budget. When the majority of the Discretionary budget is allocated to IRAD (for values less than 50% in Figure 7-6), the marginal benefit of allocating more money to Bid and Proposal is high; above 50%, the marginal benefit diminishes, but the graph still indicates that the preferred investment strategy would be to allocate somewhere between 50 to 90% of the Discretionary budget towards Bid and Proposal. When the Discretionary budget is entirely allocated to IRAD, the expected profit is \$0, because no proposals have been written that would lead to a contract award and revenue. At the opposite extreme, if the entire Discretionary budget was allocated to Bid and Proposal, there is a still significant expected profit to be made, although there is a slight dip from maximum profitability.

The trend shown in Figure 7-6 does not agree with common expectations for such a graph for an industry that produces many high technology products for the Department of Defense. Such a graph would be expected in a stable, commodity focused enterprise where marketing and branding efforts have much greater impact than research and development. It is surprising then to this is graph for an enterprise that produces advanced aerospace components. This then begs the question: why is the model attributing such low impact to research and development at TechSys?

There are two factors at play in the simulation model that contribute to this behavior. The first factor is that the time horizon for the model is simply too short. The time horizon of the model was set to three years, because this is the stated strategic outlook of TechSys, as mentioned in Section 6.2.1. That said, the benefits of research and development often take more than three years to show an impact on the profitability of the enterprise. Without extending the time horizon, the benefits of IRAD will not be apparent. An obvious test of this hypothesis is to re-run the simulation with a longer time horizon to see if the distribution of expected profitability changes to a great extent.

Figure 7-7 tests the effect of extending the time horizon from 3 years to 5 or 10 years. As can be seen from the graph, moving from 3 years to 5 years emphasizes the peak at 50% of the Discretionary budget allocated to Bid and Proposal, while slightly deemphasizing higher allocations to Bid and Proposal. As the time horizon is moved to 10 years, the effect is dramatically increased, and there is a clear preference for an even mix of investment between IRAD and Bid and Proposal. This figure indicates that TechSys should consider either extending the time horizon for the model, or keep the time horizon at three years, but understand that the long-term effects of IRAD are slightly undervalued.

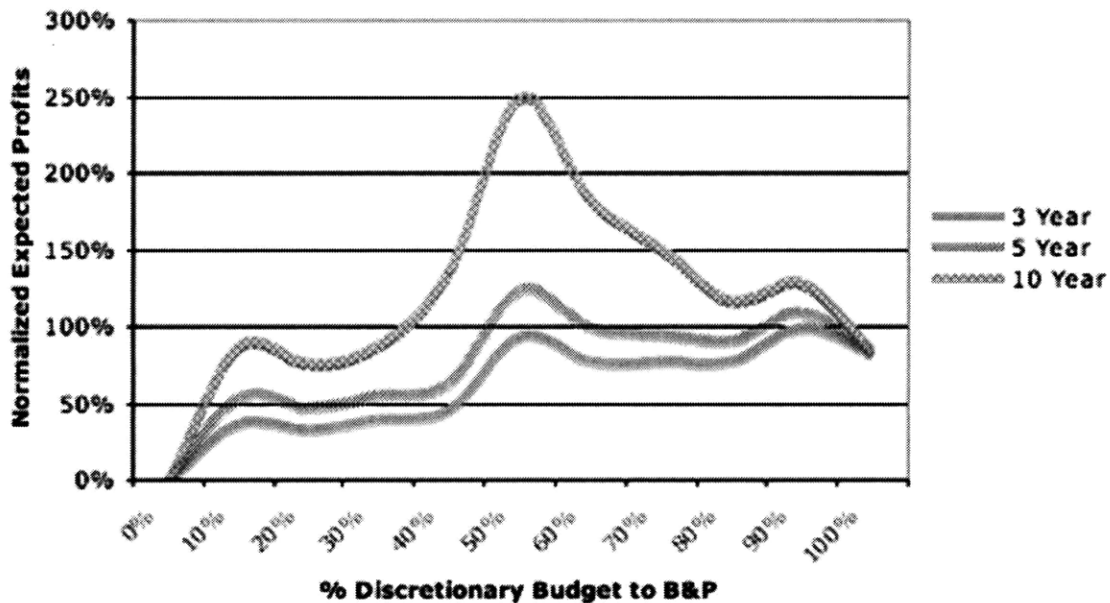


Figure 7-7: Expected profits as the Discretionary budget allocation is varied, for time horizons of 3, 5, and 10 years.

A second explanation for the underperformance of IRAD is that IRADs pursued by TechSys do not have the impact that they should. Going back to the data gathered and the construction of the model, many of those interviewed during the data collection phase of the processes commented that TechSys takes a “peanut butter” approach to investing in IRAD projects: rather than choose areas for strategic investment that is tied back to strategy, the operating units tend to

spend the money across many potential markets and technologies, without any particular focus. An internal study at TechSys in 2005 had trouble linking IRAD projects undertaken in the recent past with winning specific key proposals in that year, shedding further doubt on the effectiveness of IRAD investment at TechSys. As a result, the model randomly chooses technologies and levels of impact for each IRAD project, rather than selecting the best available from a pool of choices, as is done to select business opportunities. This diminishes the effectiveness of IRAD projects at securing competitive advantage, especially compared to pursuing new business opportunities.

Discretionary Budget Allocation Dynamics

Figure 7-8 shows a simplified causal loop diagram that captures the primary dynamics resulting from Discretionary budget allocation.

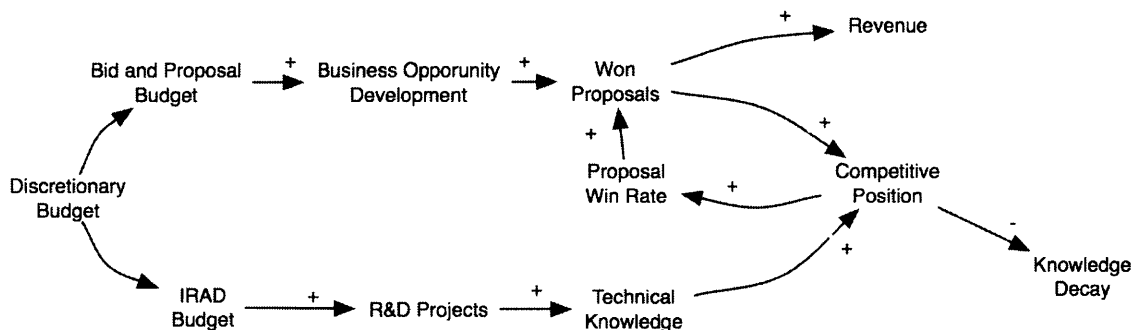


Figure 7-8: A simplified causal loop diagram of the dynamics of allocation of the Discretionary budget

As shown in Figure 7-8, both “Won Proposals” (through spending Bid and Proposal money) and “Technical Knowledge” (through spending IRAD money) can lead to increased business growth. As the competitive position increases, the probability of winning future proposals increases⁴⁰. This is balanced by a

⁴⁰This relationship is one that is often observed in the industry; contract awards in one area increase the likelihood of future awards.

“knowledge decay”: over time, the potential for capturing new business decreases, in the absence of R&D spending, since the company finds itself exploiting its existing stock of knowledge rather than creating any new knowledge and hence technical advances through R&D investment. The connection between “Won Proposals” and “Competitive Position” is short term (approximately 2 years, depending on the knowledge decay of the market), as recent success can beget future awards. The connection between “Technical Knowledge” and “Competitive Position” is longer lasting (technical knowledge is not lost).

7.2.3 Combining the Levers: The performance landscape for the current-state enterprise architecture

Thus far, the simulation model has been used to show the effect on expected profitability of varying two of the model’s input parameters: the allocation of the discretionary budget and the percentage of new proposals that are synergistic. Each analysis was performed by varying a single parameter with all others held constant. While this analysis has provided some insight into the performance characteristics of the enterprise architecture, it may be more useful to see how these parameters interact over the field of all possible combinations of inputs, as these inputs are not independent. Graphing the model’s output as both parameters are varied results in a three dimensional surface plot, with each input parameter shown on the x-axis and expected profitability shown on the y-axis. This surface can be thought of as a “performance landscape” for the enterprise, with peaks and valleys indicating the effect of different management strategies on enterprise performance. Figure 7-9 shows the performance landscape for the current state of TechSys’s enterprise architecture when the input parameters “percentage of synergy opportunities” and “Discretionary budget allocation to Bid and Proposal” are varied over the same ranges as previous analyses.

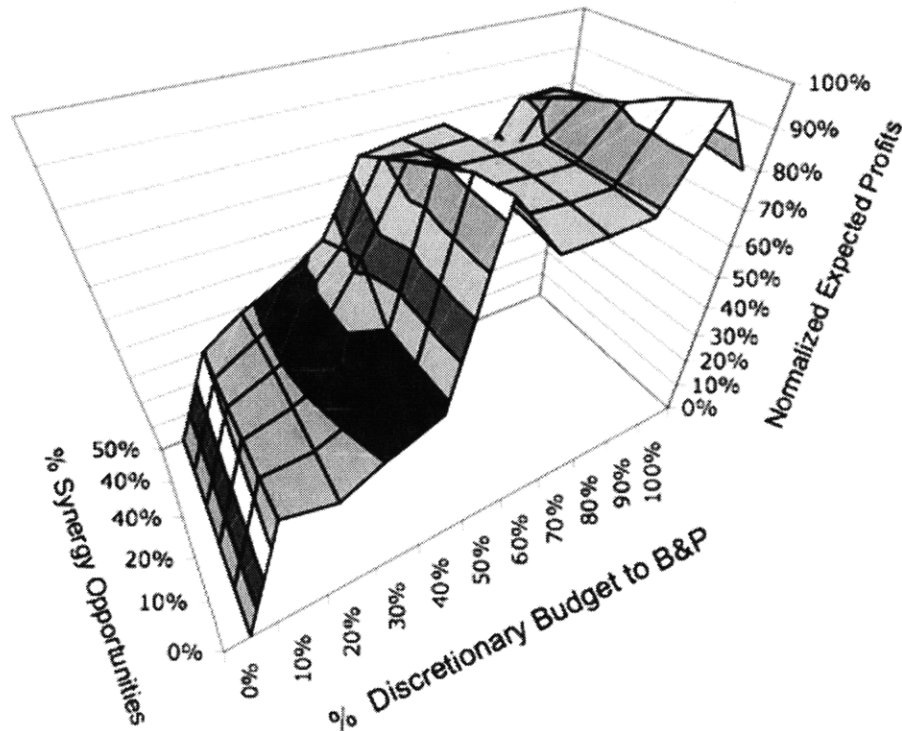


Figure 7-9: The performance landscape for the current state TechSys enterprise architecture

Looking at the performance landscape in Figure 7-9, the best possible expected profits can be achieved by not pursuing any synergy whatsoever and allocating a full 90% of the Discretionary budget to Bid and Proposal, giving IRAD only 10%. The graph suggests the most robust strategy would be to take a position on the plateau that exists between 50 to 90% of the Discretionary budget allocated to Bid and Proposal.

Many of the issues uncovered in the single variable analyses can also be seen in the performance landscape. As previously noted, as the percentage of synergy opportunities increases, profitability decreases in a monotonic manner, such that the maximum expected profitability occurs when there is no synergy whatsoever and each OU only pursues local, organic new business opportunities. As with the previous analysis of the Discretionary budget

allocation, the existing architecture favors a strategy where the majority of the budget is invested in Bid and Proposal, rather than in IRAD. Given the problems identified with the current enterprise architecture, it would not be difficult to imagine creating an alternative architecture that addresses these concerns, and is able to generate much higher expected profits.

7.2.4 Creating an Alternative Architecture

Working with TechSys stakeholders, an alternative enterprise architecture to the current state was developed to address the noted deficiencies of the current state architecture in the extremely limited sense pertaining to the allocation of available budgetary resources for capturing new business growth. The first changes to the architecture address the biases in the system against synergy. In the alternative architecture, synergy opportunities are not selected locally by the OUs, but rather by a team at the division level who select opportunities based on what has the most benefit for the division as a whole. The money for financing the new synergies will come by allocating a percentage of each OUs Bid and Proposal Budget back to the Division for synergy opportunities. The amount of this "synergy tax" on the OUs is the percentage of opportunities that have a synergy component (the same value as the model input parameter) multiplied by the OU's Bid and Proposal budget. While the division chooses how this money will be spent, the OUs will still develop the ideas, and receive the benefits from winning the resulting contracts.

The second change to the current state architecture is to select IRAD projects in a more strategically aligned fashion. This change reflects changes that were underway at TechSys at the time the simulation model was completed. In the alternative architecture, IRADs are chosen based on their expected contribution to key technology areas that are aligned with a strategic technology roadmap.

Although these roadmaps had been in use for years, they had not been used as part of the IRAD selection process.

The alternative architecture resolves both the process bias, eliminating the independent approvals needed for synergy opportunities and not for organic opportunities, and resolves the local OU incentives against choosing synergy opportunities.

The alternative architecture serves primarily as a “proof of concept” architecture, rather than as the blueprint for a future architecture that is under consideration. If the alternative architecture as described above were implemented, it would be met with widespread resistance from OUs which stand to lose a substantial portion of their Discretionary budget in the change. The alternative architecture is used here to develop a better understanding of the effects of changing the architecture, and as a starting point in future re-architecting efforts.

Table 7-1: Summary of the Alternative Architecture

-
- Created a TechSys-level oversight board to review and select synergy opportunities
 - Incentivized to choose those opportunities that benefited the overall enterprise the most
 - Synergy opportunities very closely aligned with any OU's strategic plan are selected for development
 - Created a budget category of money to fund synergy opportunities across all OUs, drawn from the OU's bid and proposal budgets evenly
 - OUs develop any synergy opportunity that TechSys has selected
 - Created new incentives for selecting IRAD projects based on their expected return and strategic alignment
-

7.2.5 Performance of the Alternative Enterprise Architecture

After coding the changes from the current architecture, the Alternative Architecture was run through the same evaluation runs from the Sections 7.3.1 and 7.3.2. Figure 7-10 shows the comparison of the current state architecture versus the alternative architecture on the dimension of investment in synergy. The data has been normalized so that 100% corresponds to the maximum expected profit under the current state architecture.

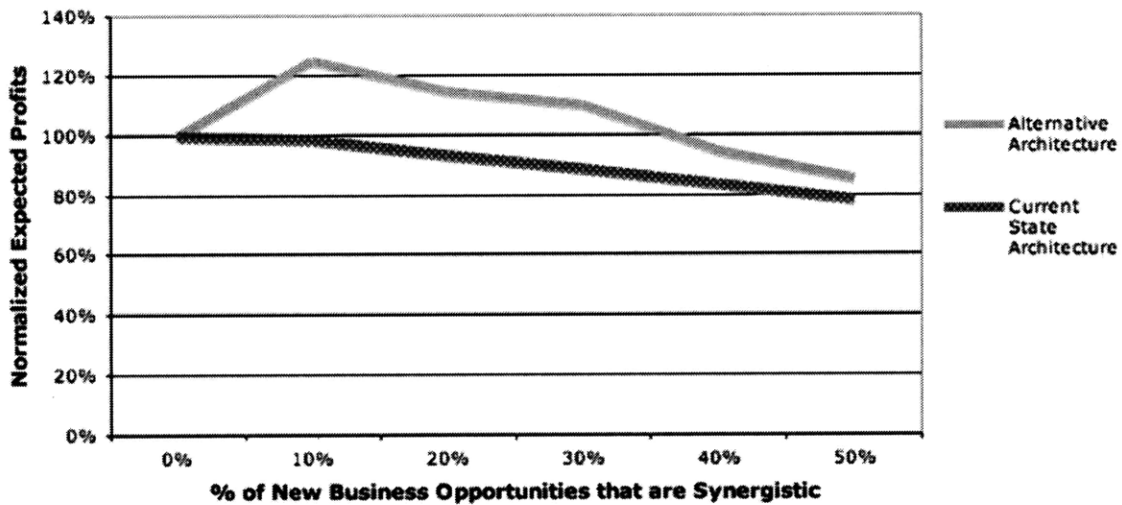


Figure 7-10: Expected profitability versus percentage of new business opportunities that are synergistic, for both the current state and alternative architectures

The first difference between the two curves is that for every point of comparison, the alternative architecture returns a greater expected profit than the current state architecture. More importantly, for many values, investing in synergy will produce an expected profit in excess of the maximum possible under the current architecture. Under the alternative architecture, synergy opportunities are able to have the impact that they were intended to have, increasing overall profitability by increasing competitive advantage and pursuing more profitable markets selling systems rather than components. At the peak of the alternative

architecture's curve, the model indicates that a 22% increase in profits is possible by moving from the current state to the alternative architecture: a change that requires only a small investment, a change in process, a change in incentives, and some amount of cultural consternation.

After peaking at 10% synergy investment, the curve trends downwards again. While it remains above the maximum possible under the current state architecture until approximately 35% synergy investment, this downward trending behavior was not anticipated. Knockout analysis, where key elements of the structure were systematically removed between model runs, was used to determine the driving structure of this downward trending dynamic. This analysis revealed that this downward trend is attributable to the fact that there are a limited number of high value synergy opportunities between the current operating units and once these high-value opportunities are exhausted, there is an opportunity cost associated with pursuing these opportunities rather than potentially more profitable organic opportunities. This misallocation of resources is due to the "walls" placed between the local OUs' Bid and Proposal budgets and the division's synergy budget, preventing a global "optimal" allocation of Bid and Proposal resources between organic and synergy opportunities across TechSys.

The relatively small number of quality synergy opportunities between operating units is attributable to the fact that there are not necessarily synergy opportunities between every pair of operating units. Of the seven operating units at TechSys, not all are in markets that could conceivably cooperate with others, causing these OUs to remain "blocked" from participating in synergy activities. In particular, one operating unit has little in common with the others, while two others have only limited potential opportunity for synergy. The potential for synergy collaboration between operating units is assessed in the model using qualitative, survey-based metrics, so there is a measure of uncertainty surrounding the exact number of potential synergy opportunities. The values used in the model should reflect a

conservative estimate of synergy opportunities. As these qualitative values are changed, the location of the peak in Figure 7-10 moves. As synergy opportunities in the model increase, the peak moves to the right and increases in amplitude. To test out the model's sensitivity, the model is re-run with the qualitative parameter that measures synergy between each OU, which assumes a value ranging from 1 to 5 on a Likert scale, is increased by 1 from its current value on the scale. As a result, the peak profitability outcome moves from its location at 10% up to 25% and increases in amplitude from 22% benefit over the base case to a 26% benefit.

Due to the sensitivity of the model to these qualitative parameters, the location and size of this peak must be evaluated with a measure of skepticism. Despite the uncertainty that exists in the graph, however, sensitivity analysis showed that the trend is robust to changes to model parameters. Even with the uncertainty as to the position and magnitude of the peak profitability level, the alternative architecture will have an appreciable increase in profitability associated with synergy investment with a peak, followed by diminishing returns.

Figure 7-11 shows the output of the alternative architecture while varying the allocation of the Discretionary budget. The expected profit is shown for both architectures, normalized such that the maximum possible from the current state architecture is 100%. As with the previous analysis of the allocation of the Discretionary budget, the synergy investment parameter was held constant.

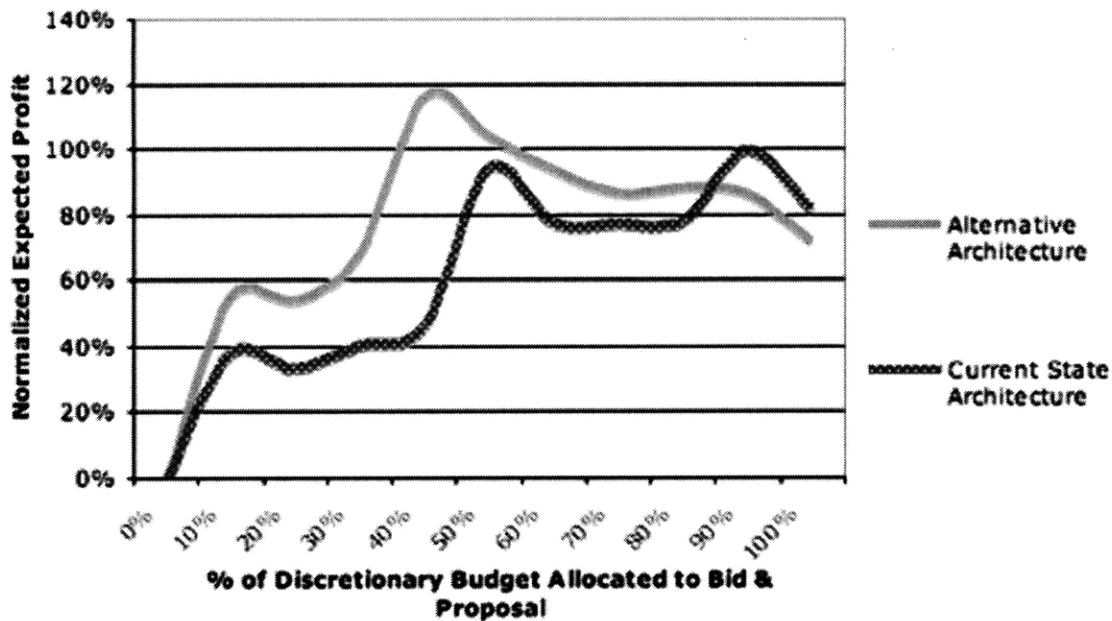


Figure 7-11: Expected Profits versus the percentage of the Discretionary budget allocated to Bid and Proposal, for both the current state and alternative architectures

As can be seen in Figure 7-11, the alternative architecture favors a more balanced approach than the current state architecture. There is a clear preference in the model to allocate approximately 40% of the Discretionary budget to Bid and Proposal, with the remainder going to IRAD. By increasing the effectiveness of IRAD through strategic selection rather than a “peanut butter” approach, simulation shows that an 18% increase in profitability can be expected. This change to the architecture also has the affect of skewing the curve to the left, giving more weight to the value of IRAD, as expected. As with previous analyses, this curve was produced using a 3-year time horizon. When the time horizon is lengthened, the tail at the far right falls more quickly, was shown in Figure 7-7.

Figure 7-12 shows the enterprise performance landscape for the alternative architecture. As can be seen from the figure, the alternative architecture has a clear maxima which balances invest between Bid and Proposal and IRAD, and

places value in a limited investment in synergy opportunities (10% of total development resources). There is no longer a flat, stable region for higher allocations of the Discretionary budget to Bid and Proposal, as seen in Figure 7-9—this architecture has a clear maximum, with a fairly steep decline in performance as more funds are allocated to IRAD.

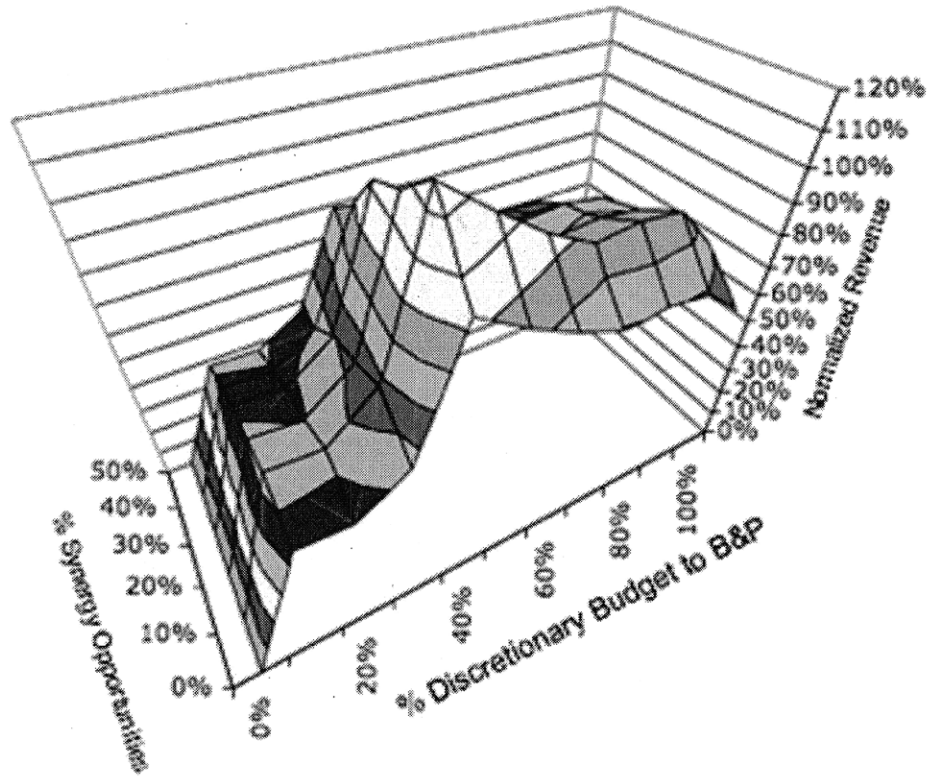


Figure 7-12: The performance landscape for the alternative enterprise architecture

7.3 RECOMMENDATIONS FROM THE USE OF THE TECHSYS SIMULATION MODEL

The TechSys Enterprise Architecture Simulation Model has proven to be a very useful tool to better understand TechSys's enterprise architecture and its effect on enterprise performance. While the model does not make "crystal ball" predictions, it is capable of being used to understand how the enterprise architecture will tend to respond to varying the control levers into the architecture. This deeper understanding of the architecture can be used to think about how the enterprise can be managed and structured going ahead, and can be used as an input when making recommendations and decisions to increase the future performance of TechSys.

At the outset of model development, there were four key questions that TechSys wanted the model to help them address, as outlined in Section 6.1.1:

1. Can TechSys achieve its growth given its current enterprise architecture with constrained resources dedicated to growth?
2. How sensitive is the architecture to changes in resource allocation?
3. What changes can be made to the architecture to improve growth opportunities given constrained resources?
4. What combination of inputs or architectural changes should be used to best grow the enterprise?

The first question is a matter of goal setting: given the current state, can we meet our goals? TechSys has established a very difficult goal to reach: doubling its net operating profit every four years, amounting to almost 20% annual growth. When looking at the performance landscape for the current state enterprise architecture in Figure 7-9, no point on the landscape can achieve TechSys's goals for profit growth. Unless TechSys almost abandons its IRAD funding and eliminates its plans to pursue synergy in an effort to maximize short-term growth,

this goal is not feasible. Even if such a drastic course were taken in order to meet the somewhat arbitrary growth goal, the competitive position of the enterprise would be severely weakened over the medium to long term due to lack of investment in the future. This approach, therefore, should be considered infeasible, and the answer to the first question should realistically be “no.” Given an anticipated decline defense spending as the American military presence in Iraq is decreased, this goal seems more and more likely to be beyond what is reasonably feasible.

Figure 7-4 (Profits as synergy investment is varied) and Figure 7-6 (Profits as the Discretionary budget allocation is varied) can be used to answer the second question: how sensitive is the architecture to changes in resource allocation? A third resource parameter used in the model, the Indirect Marketing Budget, was determined to never constrain the simulation’s performance. Much of this is attributable to the way that accounting is handled with regard to this budget (it is extremely difficult to separate what was spent on new business pursuit versus other activities given the accounting structure in place). Given the use of the Indirect Marketing Budget, it would likely follow a very similar trend to that shown for the Discretionary budget in Figure 7-7 (Expected profits as the Discretionary budget allocation is varied, for time horizons of 3, 5, and 10 years), as the money is used to fund the different points in the same process: the budgets should track very closely. The sensitivities for both synergy investment and the Discretionary budget allocation were smooth and continuous for both input parameters that were constraining and affected expected profits, with well-defined trends and maxima.

Careful analysis of these figures and their drivers in the simulation model provides a better understanding of the issues inhibiting synergy growth in the enterprise and of the effectiveness of pursuing different Discretionary budget allocation strategies. The analysis of Section 7.3.1 and 7.3.2 was used to answer the third key question: “what changes could be made to the architecture

to promote growth?” This analysis yields four key insights into understanding growth:

1. The process for selecting new business opportunities has a bias against selecting synergy opportunities;
2. The local incentives of OUs to select opportunities to fund exacerbate the barrier in the first point;
3. TechSys is not likely getting its full potential value from its IRAD activities at the present time due to lack of a process for selecting individual projects that is tied back to strategy
4. A 3 year time horizon will tend to bias against IRAD investment; Longer time horizons tend to place increased value on IRAD, evening out around the length of time it takes for an IRAD project to have its full impact.

These observations were used to design an alternative architecture described in Section 7.3.4 that avoided these barriers by changing both the process and organizational incentives for selection, moving synergy opportunity selection away from the OU level to the division level, and by increasing the value of IRADs by strategically selecting them, avoiding a “peanut butter” approach to research and development funding. This alternative architecture has the potential for up to a 21% increase in profitability over the maximum potential of the current state architecture, assuming all conditions are held equal.

The final question TechSys asked was for prescriptive guidance: “what does the simulation model tell us we need to consider changing in order to best grow the enterprise?” This question can be addressed using the enterprise landscapes for the current state and alternative architectures (Figure 7-9 and Figure 7-12). Over all inputs, the expected output of the alternative architecture exceeds that of the current state architecture. This would imply that the changes made to the current state architecture, changing the synergy selection process and the IRAD selection process, are beneficial changes that should be made to the

architecture. Given this alternative architecture, the best approach to managing it for maximum potential growth would be to have a low to modest investment in synergy (10% of all new business opportunities should be synergy), and a balanced approach to the distribution of the Discretionary budget should be used, with approximately 40-50% of the budget allocated to Bid and Proposal, with the remainder going to IRAD. Further Increasing the effectiveness of IRAD projects and discovering new ways for existing operating units to collaborate effectively (or acquire more operating units with more natural complementarities) would further increase the potential enterprise performance.

7.3.1 Other Lessons Learned from the TechSys Simulation Model

The lessons learned about the TechSys enterprise architecture as a result of model analysis are only a subset of observations made that could be beneficial to TechSys. While documenting the TechSys architecture and going through the process of creating the model, other observations were made that can provide value to the TechSys leadership.

The Knowledge View of the TechSys Enterprise Architecture

The observation with perhaps the most impact concerns the TechSys knowledge architecture. While applying the Nightingale-Rhoades Enterprise Architecture Framework (NREAF) to TechSys, a potentially significant gap was noted in the knowledge view. While changing its architecture over the past four years, TechSys had not used an enterprise architecture framework to help it consider the enterprise from multiple views. With the help of an external consultant, emphasis was placed on strategic, organizational, and process views, with little attention paid to other potential views of enterprise architecture. One view in particular that was missing was the knowledge view espoused by NREAF.

A knowledge view of the enterprise had not been employed in the past. Each OU felt that it knew the knowledge requirements of its markets well, and that these markets did not change fast enough to warrant a concerted effort by human resources or management to develop and knowledge management plan beyond basic staffing requirements. The slower tempo of changing knowledge requirements in the past did not force TechSys to explicitly create a plan to attract and develop staff with new and specific knowledge, or to move people across the enterprise to spread their expertise and help them grow their knowledge of different areas of the enterprise. With a new strategy focused on synergy between operating units in order to move from component markets to system integrator markets, new systems integration knowledge is required across all operating units in the enterprise. TechSys currently does not have strong system integration resources and knowledge, and does not have a plan for how those skills and knowledge will be acquired, developed, and shared between OUs. Considering the nature of the new direction of the enterprise, this is an omission requires attention. The simulation model did not address this aspect of the architecture because such a view was simply not developed at TechSys, but could certainly be added to the model in the future.

Open Loops

One of the early steps in creating the simulation model was to create causal loop diagrams in conjunction with TechSys stakeholders to help identify the structures that drove dynamic behavior in the enterprise architecture. In the process of creating these causal loops for the strategy view of the enterprise architecture, there were several instances of “open loops:” places where a mechanism for capturing feedback was required, but it was either weak or non-existent. Often, these critical loops, such as feedback from enterprise performance to strategic performance assessment were performed on an ad hoc basis. A similar situation exists where there is no feedback structure in place to monitor the effectiveness of strategic execution plans at meeting their intended goals. These “open loops”

were not unknown to enterprise stakeholders, but identifying them using causal loop diagramming as a tool helped them to see the importance of addressing the gaps in the architecture and helped them to see how the many aspects of strategic planning fit together on a single page. For some, the causal loop diagramming activity was the first time that many of them had seen strategic planning from a system perspective, and it served as an “ah ha” moment for several of the participants.

Enterprise Metrics

Another unanticipated benefit of modeling the enterprise architecture was the identification of potential new metrics. In order to create an executable simulation model, every aspect of the architecture must be quantified in some way. In the process of pursuing data that could be used as variables in the model, many new potential metrics were discovered, ranging from gathering statistics on IRADs and new business opportunities to identifying time constants in processes. One measure, in particular, that became very important in the model was how to measure the impact of research and development efforts. The data that was used to develop this variable in the model was extremely sparse. As Figure 7-11 shows, however, changing the effectiveness of IRAD projects can have a significant impact on resource allocation decisions. TechSys is already taking steps to develop better metrics to assess the effectiveness of its IRAD projects, and increase that effectiveness.

This example is but one of many potential metrics that could be identified because of modeling efforts. TechSys has expressed an interest in using this model in the future to identify leading indicators of performance, to help them better manage the enterprise and anticipate trends.

7.3.2 The benefits of a hybrid approach to enterprise architecture simulation modeling

Hybrid enterprise architecture simulation is a new approach to understanding enterprise dynamics, and the TechSys simulation model is the first of its kind to employ a hybrid simulation approach to analyze enterprise architecture from the multiple perspectives offered by enterprise architecture frameworks. While the previous sections have shown how this approach has been successfully applied to address the pressing concerns of TechSys, it must also be noted that some of the problems observed at TechSys, such as synergy investment behavior, could *only* be addressed through a multi-perspective approach. While hybrid simulation modeling is a useful tool in a modeler's toolbox, for some classes of problems, it is the only tool.

There were two key dynamics at play in the TechSys simulation model: Discretionary budget allocation and synergy investment. The first issue, Discretionary budget allocation, can be fairly easily captured and modeled using System Dynamics (assuming that data could be gathered that could capture aggregate characteristics of the processes). The causal loop diagram in Figure 7-8 can be expanded into a full system dynamics model by quantifying the system in terms of stocks, rates, and variables, and the extra effort of interfacing multiple models would not be required. The issue of synergy investment, however, could not be so simply captured by any one modeling approach.

The dynamics of the synergy investment problem were much more complex than the Discretionary budget problem. The problem lies at the intersection of two major views of the enterprise: process, and organization. One component of dynamics of synergy investment could be explained using a discrete event process model, similar to the one in the process sub-model of the hybrid simulation model. This doesn't tell the whole story, however. The behaviors are also *driven by the incentives of all of the operating units making locally rational*

decisions that end up producing suboptimal system-level outcomes. These dynamics were captured using the agent-based sub-model in the hybrid simulation, which treated each OU as an agent with its own decision logic guiding its funding behavior. Without both the contributions of the process sub-model as well as the organizational sub-model, the full extent of the bias against synergy opportunities in TechSys's current state architecture would not be known.

Due to the flexibility and scope of the TechSys simulation model, both of the major parameters driving enterprise behavior could be varied simultaneously, providing the ability to create the performance landscapes shown in Figure 7-9 and Figure 7-12. This would not have been possible without a hybrid simulation with an enterprise-level perspective.

This should not imply that hybrid modeling is the only approach that should be used to model the dynamics of enterprise architectures, but it does suggest that there is a class of problems that span the boundaries of architecture views that can only be fully addressed with this approach. Problems that can be described within the context of a single view can be modeled with a single simulation approach. Those that span views that are driven by very different behavioral dynamics (e.g., top down versus bottom up) may require a hybrid modeling approach to be applied. Without the aid of an enterprise framework to help with boundary setting and scoping, this class of problems has proven to be very difficult to detect and understand, as a mental exercise and from a simulation perspective. The application of enterprise architecture frameworks and hybrid simulation techniques to this class of problems provides an analytical approach that helps to manage the complexity of the dynamics and get to the causal structures and variables that drive enterprise behavior.

7.3.3 Future Application of the Simulation Model at TechSys

The TechSys Enterprise Architecture Simulation Model is a “proof-of-concept” model intended to demonstrate how a hybrid, enterprise architecture framework based approach to simulation could be used to address issues of importance to enterprises that no one simulation technique can adequately address. Despite its experimental nature, the simulation model shows great promise for further development that would allow it to increase its ability to address a wider range of issues and increase the confidence in its performance.

The simulation model, developed using JAVA and the AnyLogic simulation package, is modular with reusable components such as “operating units”, “IRAD process” and “financial functions” that can be easily linked together in a visual environment. For example, if TechSys desired to analyze the effect of a future acquisition of an additional operating unit on enterprise performance, they could insert a new OU component onto the main simulation diagram. Once the organizational and knowledge variables of the OU are assigned, the simulation model can be re-run to compare a before and after state.

Several individuals at TechSys have expressed an interest in continuing to develop the simulation model. Future work may include increasing the fidelity of the qualitative inputs that were developed via a small survey of experts and developing new metrics that could be used to better tune the model. The model can be further enhanced by expanding the limited variables representing the knowledge view into a separate sub-model. With this sub-model in place, the simulation could be used to answer a new range of questions regarding knowledge requirements and the effectiveness of different approaches to managing knowledge.

In addition to its modular characteristics, the simulation model was created with the most generic reusable components possible, allowing them to be reused by other enterprises wishing to address similar questions related to their own enterprise architecture. These components can be easily customized to a

specific enterprise by rewriting the logic that governs selection decisions, local incentives, process timing, process statistics, and more. The basic modules of the simulation were created to address the specific structures and dynamics of allocation decisions relevant to enterprises working within the financial structure required by the Defense Contracts and Auditing Agency. Fortunately, there are a large number of enterprises that fall into this category and would find such a model very useful in their own internal analyses. Despite its being largely a proof-of-concept model, the TechSys enterprise architecture simulation model is already capable of being extended to address a wider range of questions at TechSys and elsewhere within the defense and aerospace industry.

Chapter 8: CONCLUSIONS, CONTRIBUTIONS, AND FUTURE WORK

Enterprise leaders are faced with many challenges as they guide their enterprise toward its goals. They face a complex environment full of myriad components, structures, and agents that interact to produce a wide range of possible behaviors. Amidst the uncertainty of this complex environment, enterprise leaders need tools to help them better understand how their enterprise's fundamental configuration—its architecture—influences its behavior. This research has developed and applied a new analytical approach, using hybrid models, capable of simulating the dynamics of an enterprise's architecture, by explicitly linking together the behavior of the enterprise's various constituent components or domains represented in the form of multiple views. This simulation capability does not provide "crystal ball" forecasts of enterprise performance, but rather helps decision makers to understand the range of possible behaviors and performance outcomes that an enterprise architecture can produce and to enable them to gain deeper insight into how they can more effectively manage their enterprises.

As demonstrated by the TechSys case study, this approach is capable of helping enterprise leaders to examine the structure, interactions and behavior of their enterprise from new perspectives. It enables them to better understand the drivers of their enterprise's performance, identify key levers into the architecture to effect change, and it can be used to help them architect the future state of their enterprise. The TechSys case study demonstrated how an understanding of the dynamic interactions within the enterprise architecture could be used to identify

its potential performance and could be used to design a new enterprise architecture with improved performance characteristics. The TechSys hybrid simulation model was used to identify a way to increase the potential profitability of the enterprise by over 20% by making only minor changes to its process and organizational architecture without requiring additional resources for the enterprise. It is significant to note that the full extent of the bottleneck limiting TechSys's ability to execute its new business pursuit and capture process was not known until this view-spanning simulation uncovered how both the process design as well as the local incentives of operating units conspired to prevent the enterprise from pursuing joint development opportunities among its operating units, even though these "synergy" opportunities would provide more benefit to the enterprise as a whole.

8.1 REVIEW OF THESIS

The proposed approach for building hybrid simulation models of an enterprise's architecture builds upon the theoretical and applied foundations of enterprise architecting and organizational science developed in Chapter 2. Enterprise architecture frameworks were proposed as an organizing method for analysis of the enterprise as a nearly-decomposable complex system. These frameworks provide tools and guidance to partition the enterprise architecture into a set of interconnected "views," which provide abstractions of the enterprise from different and occasionally overlapping perspectives. The description and analysis of these views is informed by organizational science literature, which has developed theory-based principles and propositions to describe the relationship between an organization's design and its behavior, performance and capabilities.

Chapter 3 investigated the field of enterprise simulation to identify simulation methodologies that are capable of simulating the enterprise architecture and its multiple views while meeting the needs of enterprise leaders. Discrete event

simulation, system dynamics, and agent-based models were reviewed to build an understanding of their individual capabilities and weaknesses. This review and comparison highlighted the point that no single simulation methodology is capable of capturing the full range of behavior exhibited by an enterprise. To build this system-level analytical capability, a hybrid approach to simulating the enterprise architecture is suggested where individual enterprise architecture views are matched to a simulation methodology to form a sub-model of that view. The views should then be interconnected at the enterprise level to create a hybrid simulation model formed along the boundaries and interactions defined by an enterprise's architecture.

Chapter 4 advanced this concept by developing a generalized, iterative process and guiding principles for creating hybrid simulation models of enterprise architecture. This process provides guidance to recognizing problems for which a hybrid simulation approach is appropriate, and guides model development beginning with the documentation of the enterprise architecture, problem articulation and development of an architecturally focused hypothesis. The process then provides guidance on downselecting views in the enterprise architecture and matching them with simulation methodologies, establishing boundaries, interactions, and developing diagrams of the high-level structure of the model. After gathering data and implementing the model, a collection of approaches for testing and evaluating the model for its intended purpose is suggested, and several ways that the model may be used for enterprise architecture analysis is discussed.

Chapters 5 and 6 described the application of this process in a case study of an aerospace company called "TechSys." TechSys is a multi-divisional enterprise that sought to develop a better understanding of how its enterprise architecture supported its strategic goals to better integrate new project development across its operating units. Specifically, the company's leadership desired to know if their current state enterprise architecture was adequate to meet their strategic goals.

Each step in the modeling process developed in Chapter 4 was applied to create a working hybrid simulation model of TechSys's behaviors surrounding new business development, based upon its enterprise architecture. Chapter 7 then showed how this simulation model could be used to evaluate many aspects of TechSys's enterprise architecture and different management strategies, including:

- Process performance and design;
- Effects of local incentives for decision making;
- The allocation of resources between research and development and new business development;
- The emphasis on pursuing joint business development activities across operating units versus solely within operating units; and
- Design of alternative future states for the enterprise architecture

Most importantly, the TechSys enterprise architecture simulation model was used to demonstrate that given its current state enterprise architecture, TechSys would not be able to meet its strategic growth goals for any combination of inputs or management strategy. The model was then used to develop an alternative future state enterprise architecture with improved performance characteristics with regards to TechSys's growth goals. The model was then used to show that fairly simple changes to the architecture (simple process redesign, a slight change to the organizational structure and incentives for local decision making) could have a dramatic impact on TechSys's performance—a far greater effect than a new approach to managing and allocating resources could have. This case study clearly demonstrated the potential of hybrid simulation modeling of enterprise architecture, and paved the way for future development of this tool for enterprise leaders and enterprise architects.

8.2 CONCLUSIONS

The primary research objective of this thesis was to develop and demonstrate how a hybrid simulation model of enterprise behavior, based upon an enterprise's documented architecture, can provide insight into the linkage between an enterprise's architecture and its behavior. The application of this approach using the TechSys case study demonstrated this capability in a real-world, practical environment. There are three areas, in particular, that contributed to the successful application of this approach: the application of an enterprise architecture framework for decomposing the enterprise, in the use of a hybrid simulation approach, and in the use of a rigorous modeling process for creating the simulation model.

The use of an enterprise architecture framework to decompose the enterprise proved to be a very valuable tool for partitioning the enterprise in ways that enabled effective simulation. The framework provided valuable guidance when structuring the hybrid model, and helped to identify key boundaries and interfaces. A generalized reference model helped identify any "gaps" in the simulation model, and ensured that the architecture spanned the problem space. As discussed in the previous chapter, the Nightingale and Rhodes Enterprise Architecture Framework was able to highlight the gap in TechSys's enterprise architecture that would have linked the strategy, process and product views of its architecture with a knowledge view. This view is essential to help ensure that TechSys has the internal knowledge and capabilities necessary to execute their strategy and implement their processes. Without it, the potential for disconnects between strategy and capability rises.

The use of hybrid simulation modeling complemented the multi-view enterprise architecture by explicitly linking the structure of the enterprise to its behavior. The hybrid simulation approach was able to model each view using a simulation methodology matched to its context, be it macro-, micro-, micro-to-macro, aggregate or heterogeneous. Further, the hybrid simulation model can be

operated to quickly evaluate the effect of varying model inputs or changing the architecture, enabling hypothesis testing, scenario analysis, and sensitivity analysis of the architecture. Using a simulation model of enterprise architecture enables direct comparison of strategies and competing architectures in a way that is not possible with other enterprise architecture analysis approaches.

The hybrid simulation model developed for TechSys was able to satisfy the criteria for the utility of simulations for enterprise leaders developed in Chapter 3. The model created for TechSys was representative of the actual enterprise, its structure, and dynamics; it was able to capture behavioral complexity arising from TechSys's architecture; it addressed specific problems in a communicative and timely manner, and was adapted to facilitate hypothesis testing and scenario analysis. The area of performance that could use the most improvement is its timeliness. It took over six months to develop the hybrid simulation model after the enterprise architecture had been documented. This time could be shortened with more experience with hybrid modeling and the AnyLogic™ software, and shortened still with the development of libraries of generic enterprise simulation components.

The final aspect of this research that proved valuable was the use of a rigorous process for developing the hybrid simulation model. The use of this process forces the modeler to strictly adhere to the constructs of the enterprise framework and its boundaries and interfaces and apply these in the model in a systematic fashion. This is critical when creating models with a large number of interactions and interfaces. Without adhering to a strict process, there is a high chance of rework in the modeling process. The process helps to increase the quality of the modeling process, helping to ensure that the model is created correctly the first time.

8.3 CONTRIBUTIONS

The use of a hybrid simulation model of enterprise architecture is a new and useful contribution to both the practice of enterprise architecting and management. Over the past five years, several researchers have developed hybrid simulation models of specific enterprise processes, such as supply chain management (Scheritz and Größler 2003; Rabelo et. al. 2007) production planning (Venkateswaran and Son 2005), and manufacturing decision-making (Rabelo, et. al. 2005). These hybrid simulations have been limited in scope, however, and have not truly spanned the enterprise.

The key methodological contribution of this research was to marry the concept of hybrid simulation modeling to an enterprise architecture framework, which could be used to decompose the enterprise into a series of interconnected yet contextually unique views. The decomposition serves as the boundaries and interfaces of the hybrid simulation model. See Figure 8-1, repeated from Figure 1-2. The benefits of this approach are that many enterprises already have an enterprise architecture, or are planning on developing one, and that enterprise architecting provides a structured approach to the general analysis of an enterprise's strategy, structure, environment, and interactions.

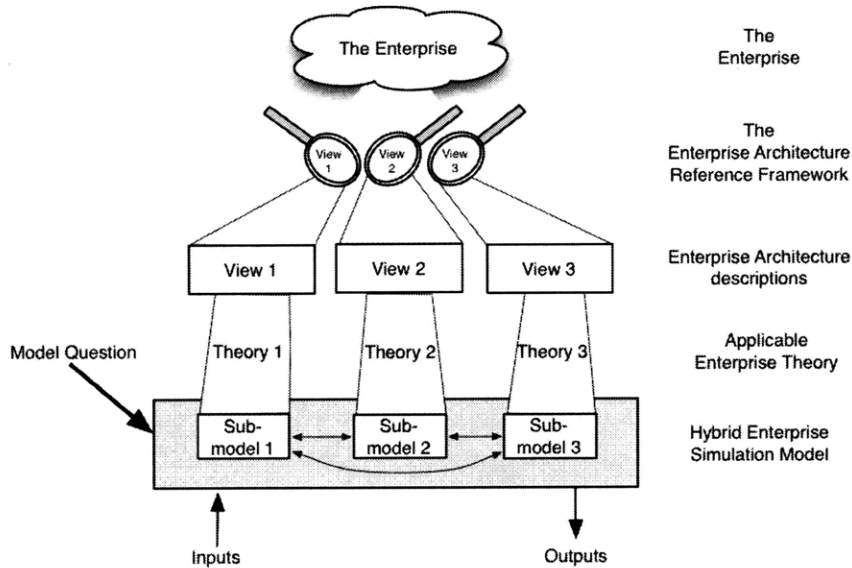


Figure 8-1: The proposed method for creating hybrid, enterprise architecture based simulation models

While enterprise architectures are currently modeled using static descriptive approaches such as UML and IDEF, developing hybrid simulation models based on the architecture greatly increases the utility of enterprise architecture as a decision tool for senior management. Today, enterprise architecting efforts that lack this simulation capability are typically used to plan and develop information systems; they emphasize the development and interactions of a technical system. Currently, the practice of enterprise architecting exists under the authority of the chief information officer of the enterprise, and is rarely, if ever, considered by the chief executive officer. Despite the admonitions of enterprise architects as to its enterprise-wide applicability, senior leadership has often overlooked enterprise architecture with its complicated “wire diagrams” of the enterprise as a useful tool that can help them manage the enterprise.

By providing integrated simulation capabilities to address all of the views of the enterprise architecture (e.g., strategy, organization, and process), to the extent that enterprise architecture is conceptualized in terms of multiple views, enterprise architecture has the potential to become a much more useful tool to the CEO and other senior leadership. Simulation modeling enables decision

makers to understand the effect of the architecture and potential changes to the architecture on the overall performance of the enterprise. Such a simulation approach provides a capability to perform sensitivity analyses, providing insight into what types of changes to the architecture can have the greatest impact on enterprise performance. Enterprise leaders can use the hybrid simulation modeling approach to understand how different views of the architecture -- from strategy and knowledge to organization and information --interact and influence each other to affect enterprise dynamics, thus being able to build "virtual experience" in developing their systems-thinking capabilities (Fowler 2003).

While not intended to serve as a forecasting tool to provide point predictions of the enterprise's future performance, the type of simulation modeling developed in this thesis is capable of showing the range of possible behaviors associated with a given enterprise architecture, as demonstrated in TechSys case study. The extremes of performance can be identified, as well as "most likely" cases. Simulation modeling of enterprise architecture can be used to compare two different candidate architectures, and through sensitivity analysis, identify key policy levers within the architecture.

Further, this research has contributed an approach to enterprise architecture simulation that should be highly flexible and extensible. This approach provides the modeler with a very open approach to create these simulation models, allowing flexibility in selecting the appropriate enterprise architecture framework and which specific simulation methodologies to use. This approach can be scaled to address problems of varying scope; what is important to note is that the enterprise architecture perspective provides a holistic, unified, framework to capture the structure and dynamics of the enterprise's complex behavior resulting from the interaction among its multiple views. Further, this approach should be able to easily accommodate new simulation approaches as they become available.

8.3.1 Other Applications of Hybrid Enterprise Architecture Simulation Models

The useful lifespan of a hybrid enterprise architecture simulation model is not limited to a single application to solve a specific problem. After it has served its original purpose, a hybrid enterprise architecture simulation model can continue to be used as a decision making aid and as a learning tool. As the enterprise's environment changes, the simulation model can be updated and used to test hypotheses about how these changes may affect the enterprise's performance. The model could be used, for example, to see what changes in the enterprise's architecture or strategies may be possible to make in order to mitigate the potentially negative effects of such environmental changes and to seize upon new opportunities offered by such changes. The simulation model can also be used as a tool to communicate to others how the enterprise functions (or should function).

Creating the simulation model in a modular fashion, as has been advocated, will allow the most flexibility for future reuse of the model. A modular design allows sub-models to be added or subtracted from the hybrid model so that the model can be expanded for use to investigate other types of enterprise behavior driven by the enterprise architecture.

8.4 DIRECTIONS FOR FUTURE WORK

This thesis has only opened the door on a new approach for analyzing the relationship of an enterprise's architecture to its behavior and for improving the ability of enterprise leaders to think about their enterprises more holistically. There is a significant amount of work remaining to develop this research into a mature simulation approach that can be easily applied to a wide array of enterprises. Foremost among future work, more case studies must be conducted to evaluate the utility of this approach over a broad range of enterprises and problems. While the TechSys case study possessed sufficient complexity that

the value of the approach could be demonstrated, more applications in a greater number of industries and for different business models (e.g., service industries, manufacturing, financial, non-profits, networked enterprises, etc.) are required to build confidence in the approach and learn of potential pitfalls in its application.

Another area for continued work is in the development of enterprise architecture frameworks at a deeper theoretical level that support the analysis of enterprises for the purposes of enterprise management and decision-making on a strategic scale, as opposed to tactical development and implementation of systems and processes. Such frameworks treat the enterprise as nearly-decomposable, complex systems with focus on the interaction among key enterprise components and the enterprise dynamics that this generates. The practice of enterprise architecting has been steadily evolving in this direction over the past decade, as an increasing number of enterprise architects have discovered the utility of enterprise architecture frameworks when applied to gain a broader perspective of alignment and interaction across the enterprise. While the Nightingale and Rhodes Enterprise Architecture Framework may be the first framework with an executive management audience in mind, the field as a whole must move in this direction by directing more research towards understanding the generalized interactions among the various enterprise views, and how different abstractions of these views and their interactions can help develop an improved understanding of enterprise dynamics. A proper conceptual framework, even without the development of a simulation model, can provide new insights for decision makers. The ability to simulate the enterprise architecture provides an additional analytical capability that can be applied for the most difficult to understand enterprise dynamics.

The final area for future work is in further developing and refining this modeling process to make it accessible to a wide population of enterprise architects and modelers. This can be done by developing a body of best practices and lessons learned to speed the development of such simulation models. These lessons

learned could be incorporated into the process to speed model development by providing, for example, templates for data collection, specific variables and interfaces that have proven critical in past applications, as well as by providing libraries of common structures, processes and functions that can be reused as parts of a hybrid simulation toolkit. Such libraries could include generic simulation models for supply chains, knowledge management, common configurations of information systems, such as enterprise resource planning systems, and customer relationship management systems. These library simulation components, each with predefined interfaces, could then be customized to particular applications. This would greatly speed up the development of these simulation models, and reduce one of the largest barriers to their adoption in practice.

8.5 CONCLUDING REMARKS

This research has developed a new and novel approach to developing models of architecture-based enterprise behavior, but much work remains to fully mature it and realize its full potential. The continued advancement of theory-driven enterprise architecture frameworks and models intended for senior management use, as well as the experience gained through repeated application of the type of simulation modeling approach developed in this thesis over a broad class of enterprises, will increase the value of such an approach as an analytical tool, and will hopefully enable enterprises to be more effectively managed and guided through the complexities that they face. The continued application of rigorous and theory-supported analytical techniques will eventually bring the management and design of enterprises from an art form mastered by few to an increasingly science-driven approach that can be more more broadly understood, valued and implemented to improve the management of complex enterprises.

1

WORKS CITED

Aerts, A. T. M., N. B. Szirbik, et al. (2002). A flexible, agent-based ICT architecture for virtual enterprises, Elsevier Science Publishers B. V.

Afuah, A. (2001). "Dynamic Boundaries of the Firm: Are Firms Better off Being Vertically Integrated in the Face of Technological Change?" Academy of Management Journal **44**(6): 1211-1228.

Anderson, P., A. Meyer, et al. (1999). "Introduction to the Special Issue: Applications of Complexity Theory to Organizational Science." Organization Science **10**(3): 233-236.

Armour, F. J., S. H. Kaisler, et al. (1999). "Building an Enterprise Architecture Step by Step." IT Pro July/August 1999: 31-38.

Axelrod, R. (1997). Advancing the Art of Simulation in the Social Sciences. Simulating Social Phenomena. R. Conte, R. Hegselmann and P. Terna. Berlin, Springer: 21-40.

Axelrod, R. (2000). Six Advances in Cooperation Theory. Ann Arbor, MI.

Axtell, R. L. (2002). Non-Cooperative Dynamics of Multi-Agent Teams. Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy.

Barabási, A.-L., E. Dezsó, et al. (2003). Scale-Free and hierarchical structures in complex networks. Modeling of Complex Systems: Seventh Granada Lectures, Spain (2002). Melville, New York, AIP.

Barki, H. and A. Pinsonneault (2005). "A Model of Organizational Integration, Implementation Effort, and Performance." Organizational Science **16**(2): 165-179.

Barney, J., M. Wright, et al. (2001). "The resource-based view of the firm: Ten years after 1991." Journal of Management **27**(6): 625-641.

Barney, J. B. (2001). "Resource-based theories of competitive advantage: A ten-year retrospective on the resource-based view." Journal of Management **27**(6): 643--650.

Barr, J. and F. Saraceno (2002). "A Computational Theory of the Firm." Journal of Economic Behavior & Organization **49**: 345-361.

Bernus, P. (2003). "Enterprise models for enterprise architecture and ISO9000:2000." Annual Reviews in Control **27**: 211-220.

Bernus, P. and L. Nemes (1999). Generalized Enterprise Reference Architecture and Methodology Version 1.6.3, IFIP-IFAC Task Force on Architectures for Enterprise Integration: 30.

Bernus, P., L. Nemes, et al. (2003). The Handbook of Enterprise Architecture, Springer.

Birkholzer, T. and J. Vaupel (2002). Handling the Complexity of IT Environments with Enterprise Architecture. Collaborative Business Ecosystems and Virtual Enterprises. M. C.-M. Luis, Kluwer Academic Publishers: 27-35.

Boccaletti, S., V. Latora, et al. (2006). "Complex Networks: Structure and Dynamics." Physics Reports **424**: 175-308.

Borschsev, A. and A. Fillipov (2004). From System Dynamics and Discrete Event to Practical Agent Based Modeling Reasons Techniques, Tools. Proceedings of the 22nd International Conference of the System Dynamics Society, Oxford, England.

Bozdogan, K. (2004). Enterprise Architecting Knowledge Area Research Agenda, Massachusetts Institute of Technology, Lean Aerospace Initiative.

Bryjolfsson, E. and L. Hitt (2003). Beyond Computation: Information Technology, Organizational Transformation, and Business Performance. Inventing Organizations of the 21st Century. T. W. Malone, R. Laubacher and M. S. Morton. Cambridge, MIT Press.

Burke, W. W. and G. H. Litwin (1992). "A Causal Model of Organizational Performance and Change." Journal of Management **18**(3): 523-545.

WORKS CITED

Carley, K. M. (1995). "Computational and Mathematical Organization Theory: Perspective and Directions." Computational and Mathematical Organization Theory 1(1): 39-56.

Carley, K. M. (2002). "Computational organizational science and organizational engineering." Simulation Modelling Practice and Theory 10: 253-269.

Carley, K. M. and L. Gasser (1999). Computational Organization Theory. Multiagent Systems. G. Weiss. Cambridge, MIT Press.

Carroll, T. N., T. J. Gormley, et al. (2006). "Designing a New Organization at NASA: An Organization Design Process Using Simulation." Organization Science 17(2): 202-214.

Chang, M.-H. and J. E. Harrington Jr. (2006). Agent-Based Models of Organizations. Handbook of Computational Economics II: Agent-based Computational Economics. K. L. Judd and L. Testafion, North-Holland.

Checkland, P. (1981). Systems Thinking, Systems Practice. New York, Wiley Press.

Child, J. and R. G. McGrath (2001). "Organizations Unfettered: Organizational Form in an Information-Intensive Economy." Academy of Management Journal 44(6): 1135-1148.

Coase, R. H. (1937). "The Nature of the Firm." Economica 4(16): 386-404.

Crowston, K. (2003). A Taxonomy of Organizational Dependencies and Coordination Mechanisms. T. W. Malone, K. Crowston and G. A. Herman, MIT Press: 85-108.

Davis, J. P., K. M. Eisenhardt, et al. (2007). "Developing Theory through Simulation Methods." Academy of Management Review 32(2): 480-499.

Dodds, P. S., D. J. Watts, et al. (2003). "Information exchange and the robustness of organizational networks." PNAS 100(21): 12516-12521.

Dooley, K. J. and A. van de Ven (1999). "Explaining Complex Organizational Dynamics." Organization Science 10(3): 358-372.

Drucker, P. (1988). "The Coming of the New Organization." Harvard Business Review **66**(1): 45-53.

Dyer, J. H., P. Kale, et al. (2001). "How to Make Strategic Alliances Work." Sloan Management Review **42**(4).

Epstien, J. (2003). Growing Adaptive Organizations: An Agent-Based Computational Approach, The Brookings Institution.

Ethiraj, S. K. and D. Levinthal (2004). "Bounded Rationality and the Search for Organizational Architecture: An Evolutionary Perspective on the Design of Organizations and their Evolvability." Administrative Science Quarterly **49**: 404-437.

Ethiraj, S. K. and D. Levinthal (2004). "Modularity and Innovation in Complex Systems." Management Science **50**(2): 159-173.

Evans, S. and N. Roth (2004). Collaborative Knowledge Networks. Collaborative Networked Organizations: A Research Agenda for Emerging Business Models. L. Camarinha-Matos and H. Afsarmanesh. Boston, Kluwer Academic Publishers: 153-170.

Forrester, J. (1958). "Industrial Dynamics: A Major Breakthrough for Decision Makers." Harvard Business Review **36**(4): 37-66.

Fowler, A. (2003). "Systems Modelling, simulation, and the dynamics of strategy." Journal of Business Research **56**: 135-144.

Fox, M. S., M. Barbuceanu, et al. (1995). An Organisation Ontology for Enterprise Modelling: Preliminary Concepts for Linking Structure and Behaviour. 4th Annual Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE'95).

Fox, M. S. and M. Gruninger (1998). Enterprise Modeling. Artificial Intelligence Magazine, American Association for Artificial Intelligence. **Fall 1998**: 109-122.

Frey, D. D. and X. Li (2004). Evaluating Robust Design Methods Using a Model of Interactions in Complex Systems. Engineering Systems Symposium, MIT.

WORKS CITED

Galbraith, J. R. (1973). Designing Complex Organizations. Reading, MA, Addison-Wesley Publishing Company.

Galbraith, J. R., D. Downey, et al. (2002). Designing dynamic organizations: a hands-on guide for leaders at all levels. New York, AMACOM.

George, A. L. and A. Bennett (2005). Case Studies and Theory Development in the Social Sciences. Cambridge, MIT Press.

Gharajedaghi, J. (1999). Systems Thinking: Managing Chaos and Complexity: A Platform for Designing Business Architecture. Boston, Butterworth-Heinemann.

Glazner, C. (2006). Enterprise Integration Strategies Across Virtual Extended Enterprise Networks: A Case Study of the F-35 Joint Strike Fighter Program Enterprise. Technology and Policy Program. Cambridge, Massachusetts Institute of Technology. **S.M.:** 184.

Guyot, P. and S. Honiden (2006). "Agent-Based Participatory Simulations: Merging Multi-Agent Systems and Role Playing Games." Journal of Artificial Societies and Social Simulation **9(4)**: 8.

Harding, J. A., B. Yu, et al. (1999). "Informational modelling: an integration of views of a manufacturing enterprise." Int. J. Prod. Res. **37(12)**: 2777-2792.

Hax, A. and N. S. Majluf (1981). "Organizational Design: A Survey and an Approach." Operations Research **29(3)**: 417-447.

Hofstadter, D. (1979). Gödel, Escher, Bach: an Eternal Golden Braid. New York, Basic Books.

Holland, J. H. (1992). "Complex Adaptive Systems." Daedalus **121(1)**: 17-30.

Kalpic, B. and P. Bernus (2002). "Business process modelling in industry--the powerful tool in enterprise management." Computers in Industry **47**: 299-318.

Kates, A. (1999). "Introduction to the Special Issue: Applications of Complexity Theory to Organization Science." Organization Science **10(3)**: 223-236.

Keller-McNulty, S., et al. (2006). Defense Modeling, Simulation, and Analysis: Meeting the Challenge. Washington, D.C., National Academies Press: 86.

Koen, B. V. (2003). Discussion of the Method. Oxford, Oxford University Press.

Krutchen, P. (1995). "Architectural Blueprints--The "4+1" View Model of Software Architecture." IEEE Software **12**(6): 42-50.

Kunz, J. C., R. E. Levitt, et al. (1988). "The Virtual Design Team: A Computational Simulation Model of Project Organization." Communications of the Association for Computing Machinery **41**(11): 84-91.

Kuras, M. L. and B. E. White (2005). Engineering Enterprises Using Complex-System Engineering. INCOSE White Paper.

Lewin, A. Y., C. P. Long, et al. (1999). "The Coevolution of New Organizational Forms." Organizational Science **10**(5): 535-550.

Lim, S. H., N. Juster, et al. (1997). "Enterprise Modelling and Integration: A Taxonomy of Seven Key Aspects." Computers in Industry **34**: 339-359.

Lim, S. H., N. Juster, et al. (1997). "The Seven Major Aspects of Enterprise Modelling and Integration: A Position Paper." ACM SIGGROUP Bulletin **18**(1): 71-75.

Lin, F.-R., G. W. Tan, et al. (1999). "Multiagent Enterprise Modeling." Journal of Organizational Computing and Electronic Commerce **9**(1): 7-32.

Lomi, A. and E. R. Larsen, Eds. (2001). Dynamics of Organizations: Computational Modeling and Organizational Theories. Cambridge, MA, MIT Press.

Luckham, D. (2002). The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems, Addison-Wesley Professional.

Madarasz, L., M. Timko, et al. (2004). Enterprise Modeling and its Applications in Company Management Systems. 5th International Symposium of Hungarian Researchers on Computational Intelligence, Budapest.

WORKS CITED

Madnick, S. and R. Wang (1988). A Framework of Composite Information Systems for Strategic Advantage. Proceedings of the 1988 Hawaii International Conference on System Sciences.

Magretta, J. (2002). "Why Business Models Matter." Harvard Business Review **80**(5): 86-93.

Mahidhar, V. (2005). Designing the Lean Enterprise Performance Measurement System. Engineering Systems Division. Cambridge, Massachusetts Institute of Technology. **S.M.**

Maier, M. W. (1999). "Architecting Principles for Systems-of-systems." Systems Engineering **1**: 267-284.

Malone, T., R. Laubacher, et al., Eds. (2003). Inventing the Organizations of the 21st Century. Cambridge, MIT Press.

Malone, T. W. and J. F. Rockart (1991). Computers, Networks, and the Corporation. Scientific American. **265**: 128-136.

March, J. G. (1991). "Exploration and Exploitation in Organizational Learning." Organization Science **2**(1): 71-87.

McKelvey, W. (1999). "Avoiding Complexity Catastrophe in Coevolutionary Pockets: Strategies for Rugged Landscapes." Organization Science **10**(3): 294-321.

Milgrom, P. and J. Roberts (1995). "Complementarities and fit Strategy, Structure, and Organizational Change in Manufacturing." Journal of Accounting and Economics **19**: 179-208.

Miller, J. H. (1998). "Active Nonlinear Tests (ANTs) of Complex Simulation Models." Management Science **44**(6): 820-830.

Mingers, J. and A. Gill (1997). Multimethodology: Towards a theory and practice of combining management science methodologies. Chichester, John Wiley & Sons.

Moldoveanu, M. C. and R. M. Bauer (2004). "On the Relationship Between

Mostashari, A and J Sussman (2009), "A Framework for Analysis, Design and Management of Complex Large-Scale Interconnected Open Sociotechnical Systems", International Journal of Decision Support System Technology, 1(2):

Organizational Complexity and Organizational Structuration." Organizational Science 15(1): 98-118.

Morel, B. and R. Ramanujam (1999). "Through the looking glass of complexity: The dynamics of organizations as adaptive and evolving systems." Organization Science 10(3): 278-293.

Murman, E., et al. (2002). Lean Enterprise Value: Insights from MIT's Lean Aerospace Initiative. New York, Palgrave.

Nightingale, D. and D. H. Rhodes (2004). Enterprise Systems Architecting: Emerging Art and Science within Engineering Systems. Engineering Systems Symposium. Cambridge, MA.

Nonaka, I., T. Ryoko, et al. (2000). "SECI, Ba and Leadership: a Unified Model of Dynamic Knowledge Creation." Long Range Planning 33(1): 5--34.

Oliver, D., T. Kelliher, et al. (1997). Engineering Complex Systems with Models and Objects. Washington, D.C., McGraw-Hill.

Ouzounis, E. K. (2001). An Agent-Based Platform of the Management of Dynamic Virtual Enterprises. Von der Fakultät IV – Elektrotechnik und Informatik der Technische Universität Berlin zur Erlangung des akademischen Grades. Berlin, TU Berlin. **Doktor der Ingenieurwissenschaften**.

Pan, J. Y. C. and J. M. Tenenbaum (1991). "An Intelligent Agent Framework for Enterprise Integration." IEEE Transactions of Systems, Man, and Cybernetics 21(6): 1391.

Parise, S. and L. Sasson (2002). "Leveraging Knowledge Management Across Strategic Alliances." Ivey Business Journal.

Porter, M. (2001). "Strategy and the Internet." Harvard Business Review 79(3): 62-78.

WORKS CITED

Prietula, M. J., K. M. Carley, et al., Eds. (1998). Simulating Organizations: Computational Models of Institutions and Groups. Cambridge, MIT Press.

Rabelo, L., H. Eskandari, et al. (2007). "Value Chain Analysis using hybrid simulation and AHP." International Journal of Production Economics **105**: 536-547.

Rabelo, L., M. Helal, et al. (2005). "Enterprise Simulation: A Hybrid System Approach." International Journal of Production Economics **105**: 536-547.

Rechtin, E. (2000). Systems Architecting for Organizations. Boca Raton, CRC Press.

Rechtin, E. and M. W. Maier (1997). The Art of Systems Architecting. Boca Raton, FL, CRC Press.

Richards, M. G., N. B. Shah, et al. (2006). Managing Complexity with the Department of Defense Architecture Framework: Development of a Dynamic System Architecture Tool. INCOSE.

Rivkin, J. W. and N. Siggelkow (2003). "Balancing Search and Stability: Interdependencies Among Elements of Organizational Design." Management Science **49**(3): 290-311.

Ross, J. W., H. Weil, et al. (2006). Enterprise Architecture as Strategy. Cambridge, Harvard Business School Press.

Sawyer, R. K. (2003). "Artificial Societies: Multiagent Systems and the Micro-Macro Link in Sociological Theory." Sociological Methods & Research **31**(3): 325-363.

Schekkerman, J. (2004). How to survive in the jungle of Enterprise Architecture Frameworks. Victoria, Trafford.

Schieritz, N. and A. Größler (2003). Emergent Structures in Supply Chains: A Study Integrating Agent-Based and System Dynamics Modeling, IEEE Computer Society.

Schilling, M. A. and H. K. Steensma (2001). "The Use of Modular Organizational Forms: An Industry-Level Analysis." The Academy of Management Journal **44**(6): 1149-1168.

Scholl, H. J. (2001). Agent-based and System Dynamics Modeling: A Call for Cross Study and Joint Research. Proceedings of the 34th Annual Hawaii International Conference on System Science, IEEE- Computer Society.

Scott, W. R. and G. F. Davis (2003). Organizations and Organizing: Rational, Natural and Open System Perspectives. Upper Saddle River, NJ, Pearson Prentice Hall.

Sharman, D. M. and A. A. Yassine (2003). "Characterizing Complex Product Architectures." Systems Engineering 7(1): 35-60.

Sigglekow, N. and D. Levinthal (2003). "Divide to Conquer: Centralized, Decentralized, and Reintegrated Organizational Approaches to Exploration and Adaptation." Organization Science 14(6): 650-669.

Simon, H. A. (1969). Sciences of the Artificial. Cambridge, MA, MIT Press.

Simon, H. A. (1990). "Prediction and Prescription in Systems Modeling." Operations Research 38(1): 7-14.

Simon, H. A. (1997). "Near-Decomposability and the speed of evolution." Industrial and Corporate Change 11(3): 587-599.

Simon, H. A. and A. Ando (1961). "Aggregation of Variables in Dynamics Systems." Econometrica 29(2): 111-138.

Singleton, R. A. J. and B. C. Straits (2005). Approaches to Social Research. Oxford, Oxford University Press.

Sole, R. V. and S. Valverde (2004). Information Theory of Complex Networks: on evolution and architectural constraints. Complex Networks. H. F. a. Z. T. E. Ben-Naim, Springer, Berlin: 169-190.

Sousa, G. W. L., E. M. Van Aken, et al. (2002). "Applying an Enterprise Engineering Approach to Engineering Work: A Focus on Business Process Modeling." Engineering Management Journal 14(3): 15-24.

Sterman, J. D. (1991). A Skeptic's Guide to Computer Models. Managing a Nation: The

WORKS CITED

Microcomputer Software Catalog. G. O. Barney. Boulder, CO, Westview Press: 209-229.

Sterman, J. D. (2000). Business Dynamics: Systems Thinking and Modeling for a Complex World. Boston, Irwin McGraw-Hill.

Sterman, J. D. (2001). "System Dynamics Modeling: Tools for Learning in a Complex World." California Management Review **43**(4): 8-25.

Sterman, J. D. (2002). "All models are wrong: reflections on becoming a systems scientist." Systems Dynamics Review **18**(4): 501-531.

Sutherland, J. and W.-J. van den Heuvel (2002). "Enterprise Application Integration and Complex Adaptive Systems." Communications of the Association for Computing Machinery **45**(10): 59-64.

Thompson, J. (1967). Organizations in Action: Social Science Bases of Administrative Theory. New York, McGraw-Hill.

Tilebein, M. (2006). Principles of Emergence - A Generic Framework of Firms as Agent-Based Complex Adaptive Systems. 24th International Conference of the System Dynamics Society, Nijmegen, The Netherlands, Wiley InterScience.

Troche, C. (2006). Documenting Complex Systems in the Enterprise. International Conference on Complex Systems 2006, Boston, MA, NECSI.

Venkateswaran, J. and Y.-J. Son (2005). "Hybrid system dynamic--discrete event simulation-based architecture for hierarchical production planning." international Journal of Production Research **20**(15): 4397-4429.

Vernadat, F. (1996). Enterprise Modeling and Integration: Principals and Applications. New York, Chapman & Hall.

Weiss, G., Ed. (1999). Multiagent Systems. Cambridge, MIT Press.

White, B. (2008). Systems Science: Deepening Our Understanding of the Theory and Practice of Systems Engineering. INSIGHT, INCOSE. **11**: 5-19.

Whitman, L. E., B. L. Huff, et al. (1998). The needs and issues associated with representing and integrating multiple views of the enterprise. Information Infrastructure Systems for Manufacturing II, Chapman and Hall.

Wissmann, L. A. and A. A. Yassine (2004). Product Architecture and the Firm, Product Development Laboratory, Department of General Engineering, University of Illinois at Urbana-Campaign.

Wolfram, S. (2002). A New Kind of Science, Wolfram Media.

Yassine, A. A., N. Joglekar, et al. (2003). "Information Hiding in Product Development: The Design Churn Effect." Research in Engineering Design **14**: 145-161.

Yin, R. K. (2003). Case Study Research: Design and Methods. Thousand Oaks, CA, Sage Publications.

Yu, B., J. A. Harding, et al. (2000). "Supporting Enterprise Design Through Multiple Views." International Journal of Agile Management Systems **2**(1): 71-77.

Zachman, J. A. (1987). "A Framework for Information Systems Architecture." IBM Systems Journal **26**(3): 276-292.

Zinn, A. W. (2004). The Use of Integrated Architectures to Support Agent Based Simulation: An Initial Investigation. Graduate School of Engineering and Management, Air Force Institute of Technology. **M.S.**