2011 Progress Report:
Research on Understanding the Challenges to Net-Centric Systems and Mitigating Approaches

Stuart E. Madnick
Allen Moulton
Xitong Li
Travis Trammell

Working Paper CISL# 2012-01

January 2012

Composite Information Systems Laboratory (CISL)
Sloan School of Management, Room E62-422
Massachusetts Institute of Technology
Cambridge, MA 02142

DIA-Lincoln Laboratory-MIT Joint Research on Understanding the Challenges to Net-Centric Systems and Mitigating Approaches

Progress Report
November 1, 2010 through December 31, 2011

Stuart Madnick, Allen Moulton, Xitong Li, Travis Trammell
MIT Sloan School of Management
Cambridge, MA 02139

TABLE OF CONTENTS

The work reported herein was supported, in part, by the MIT Lincoln Laboratories and the Defense Intelligence Agency (DIA) under the "Understanding the Challenges to Net-Centric Systems and Mitigating Approaches" project, MIT Lincoln Laboratory contract 16-11-TCO-0013. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not reflect the official policy or position of MIT Lincoln Laboratory, the Defense Intelligence Agency or the Department of Defense.

PART I

Lincoln Laboratory-MIT Joint Research on Understanding the Challenges to Net-Centric Systems and Mitigating Approaches

Progress Report
November 1, 2010 through December 31, 2011

Stuart E. Madnick, Allen Moulton, Xitong Li, Travis Trammell

1. Introduction

For decades, organizations across the spectrum of private industry and government have noted the limitations to information sharing and the resulting problems caused by "stovepipe" information systems. Despite this recognition, occasional "crises" (e.g., Y2K and 9-11), and various technological and organizational efforts, these problems largely continue to persist and continue to impact information quality in the enterprise (e.g., timeliness, completeness, consistency, believability, and at times even accuracy).

Within the DOD, the net-centric doctrine has been around and official policy for nearly a decade, but the legacy systems that still exist and, to a large extent, the new systems that are built often have more of a "stovepipe" rather than "net-centric" (or SOA) character.

The research questions are:

- **why is the change from "stove-pipe" to "net-centric" proceeding so slowly** and running into roadblocks and bottlenecks even in an organization that has a hierarchical command structure with top level buy-in to the net-centric concept.
- **what mitigating technological, managerial, and organizational approaches and interventions** can be introduced to improve the speed of transition and the quality of the resultant end product systems.

An improved understanding of the dynamic process of evolutionary IT systems development will help in identifying necessary policies, incentives, and command and control structures for migrating towards the net-centric vision.

2. Outline of Research Project

The evolutionary information system development process involves the dynamic interaction of many stakeholders and processes that can lead to markedly different outcomes depending on how the overall process is managed and how incentives influence decisions made by various stakeholders. This phenomenon is related to the classic problem of incremental planning and implementation resulting in systems which deviate from the designed architecture even though each development step along the way made sense to the relevant decision-maker at the time.

The research builds on successful prior results in applying System Dynamics to the software development management process. The proposed research focus will apply the System Dynamics approach, as well as employ other related research processes (e.g., organizational theory, economics) as relevant, to modeling the roles and interactions of key stakeholders and their actions to identify management policy levers that can influence and improve the evolutionary acquisition process.

3. Work Accomplished in the 2010-2011 Period

During the initial period from 1 Nov 2010 through 31 Dec 2011, this MIT research project accomplished the following:

(a) investigated the experience both in DOD and in industry of attempts to develop net-centric systems (see Appendix:"A Brief Overview of DOD and Industry Experience with Net-Centric Systems");

(b) developed a framework of challenging and mitigating factors affecting success or failure of these systems (see Section 4); and

(c) developed two System Dynamics models for representing and exploring the effects of policy choices on outcomes. These models are:

- System Dynamics model of Management Commitment & Technology Learning Factors (see Section 5)
- System Dynamics model of the Effects of Budget Fluctuations on Software Acquisition (see Section 6)

Experiments were done with both models to better understand the dynamics and outcomes based on policy alternatives. Based on these experiments, the models were refined and adjusted to better represent realistic scenarios.

Research progress was presented at meetings held both at MIT and at DIA at intervals throughout the period. Presentation materials for meetings on the research were delivered electronically. Reports and draft academic papers were also prepared and made available. An access-controlled Wiki was built to hold resources used in the project as well as research results. The meetings held were:

- 2010-06-08 Pre-Kickoff Meeting with DIA and LL at LL
- 2010-11-16 Meeting with LL at MIT
- 2011-03-17 Meeting with DIA and LL at MIT
- 2011-05-02 Meeting with DIA and LL at MIT
- 2011-05-26 Meeting at DIA
- 2011-07-19 Meeting at DIA
- 2011-08-23 Meeting with DIA and LL at MIT
- 2011-10-20 Meeting with DIA and LL at MIT
- 2011-12-19 Meeting with DIA and LL at MIT

4. Challenging and Mitigating Factors

Based on a review of academic literature on implementation of systems similar to net-centric systems (e.g., ERP and SOA systems) as well as the case examples of DOD and industry experience described in the Appendix, the team developed an initial list of challenging and mitigating factors for net-centric success. This list of factors was then validated and extended during open-ended discussions with subject matter experts from DIA, DOD, academia, and industry. A list of 55 specific factors was identified and then analyzed and grouped into nineteen factor categories as a basis for further investigation and system dynamics modeling. These nineteen categories are:

1) Quantifying benefits
2) Unbalance between benefits and costs (over time, and across the organization)
3) Commitment trap
4) Uncertainty
5) Standardization
6) Trust and security
7) Delivery pressure
8) Cost pressure
9) Support of NC infrastructure software
10) Fuzziness of goal
11) Culture
12) Counter-politics
13) Technology limitations
14) Different requirement
15) Complexity management
16) Budget
17) Organization instability
18) IT crises as initiating events
19) Legal constraints

Further work is proceeding to develop and organize the factor list and to frame additional questions based on academic theory.

5. Model 1 - Management Commitment & Technology Learning Factors

One of the features of net-centric systems is the decomposition of whole siloed systems into sharable components using a Service-Oriented Architecture (SOA. SOA has become increasingly popular in defense systems and in industry. Some of the related programs and common characteristics are listed below:

- **Programs for Implementing Organization-wide Interoperable IT systems**
  - Service-oriented Architecture (SOA)
  - Enterprise data standardization
  - Digital reuse efforts
  - Enterprise data/information quality improvement
  - DoD “net-centric” strategy
- **Common Characteristics**
  - Intended benefits
    - Reduced delivery time, enhanced responsiveness to needs
    - Reduced development and maintenance costs
    - Increased IT development productivity
    - Improved data flow and usage
  - Require organization-wide change and top management commitment

Nevertheless, mixed outcomes have often been reported about SOA implementation. The key research question to be answered in this study is how various factors, while interacting with each other over time, affect the dynamics of the SOA implementation and certain organizational challenges may occur in the implementation process.

To develop plausible explanations for the organizational challenges to SOA implementation, this study builds a system dynamics model from a process research perspective which is missing in the extant literature on SOA. In particular, the relationship between the duration (rather than just the strength) of top management normative commitment and the time delay of technology learning, which is critical to the success of SOA implementation but ignored in extant literature, is uncovered and the critical effects on the success of SOA implementation is demonstrated through experimental simulations.

The simulations show that reducing the duration of normative commitment may cause dramatically unexpected and usually negative outcomes of SOA implementation. The implications are that when providing normative commitment on SOA implementation, managers should consider the time delay of SOA technology learning in their organization and carefully decide appropriate duration of their normative commitment.

This research adopts a process perspective to investigate the dynamics of SOA implementation, contrasting to prior SOA research using only the variance perspective. Specifically, a system dynamic model is built to unfold the significant effects of the duration of normative commitment and time delay of technology learning on the dynamics of the SOA implementation process. Although prior research had identified top management commitment as a critical success factor, this research suggests that it is the duration (rather than just the strength) of normative commitment that may dramatically affect the dynamics. Further, this research suggests that in order to reach a successful SOA implementation, it is important to understand the duration of normative commitment in relation to the technology learning time, i.e., the time to learn SOA technology. See Part II of this report for a detailed explanation.

6. Model 2 - Effects of Budget Fluctuations on Software Acquisition

This research involves a system dynamics study of the impact of funding fluctuations on government sponsored software development. Funding fluctuations are a fact of life in government funded enterprises. Additionally, top line funding fluctuations are magnified at the contract level. Some fluctuations appear as reductions and increases with the most severe cases being funding gaps with complete work stoppages. System dynamics was used to model the impact of funding fluctuation based staffing changes on software development productivity and the impact on delivery schedule and cost. The study found that funding gaps and funding reductions reduce software development productivity and delay completion much more than might be expected. Complex development projects take time to reach full productivity and suffer a productivity cost when stopped and restarted. This study also concluded that if funding has to be cut, it is less harmful to spread the cuts out rather than to have gaps. System dynamics proved to be an effective tool for evaluating the impact of funding fluctuations. See Part III of this report for a detailed explanation.

7. Conclusion and Future Work

The models described above explore several aspects of the problems affecting the acquisition of net-centric systems that will be effective in meeting the important goals stated in Objective 1.2 of the *DIA 2012-2017 Strategic Plan:* "Develop **common processes, collaborative tools**, and innovative approaches to improve **horizontal and vertical information sharing** and timely access to data."

Meeting these goals has proven difficult in the past and will likely continue to be difficult in the future. The software and the interaction of systems and people in the extended defense intelligence enterprise are inherently complex. In their study *Organizing for a Complex World*, Berteau, Ben-Ari, and Zlatnik (2009, p.2) conclude that "Complex defense and network-centric systems have proven to be difficult to develop on time and on budget, a consequence of the complexity inherent in both the systems and the acquisition environment." By modeling the process, we hope to be able to enable managers to experiment with alternative approaches to achieving the goals of information sharing in the context of defense intelligence.

As we found in our review of DOD experience and study of funding effects, the necessary regulations and constraints create significant hurdles in the path to acquiring new systems. The Defense Science Board (2009) found that "**The conventional DOD acquisition process is too long and too cumbersome to fit the needs of the many systems that require continuous changes and upgrades**. Many existing programs are exceeding cost and schedule baselines, which cannot continue unabated." (p.4)

IT systems have historically proved even more difficult to acquire than weapons systems. In Section 704 of the FY10 Defense Authorization, Congress directed DOD to study the problem and recommend new approaches. The OSD Report to Congress (2010) proposed changes where "IT will be acquired as 'time-boxed' projects delivering capability in an iterative fashion using mature technologies, while managed in capability-aligned portfolios to identify and eliminate redundancy." (p.2)

The same types of complexity and system acquisition problems also occur in industry. The National Research Council (2010) found that "[a]gile approaches and iterative, incremental approaches to software development have been receiving increased attention in industry in recent years and are having a significant impact on how software is developed and how systems are tested." (p.86)

A key question is how agile methods can be successfully applied within the context of DOD software acquisition which must follow laws and regulations governing, among other things, contracting procedures and system certification and test. However difficult the problem may be, it is not intractable and we will work to develop models to assist managers in understanding, exploring, and choosing among potential courses of action within the constraints of regulations and the interactions of stakeholders.

During 2012, building on the results of the prior work:

(a) We plan to extend and deepen our understanding of the interplay of challenging and mitigating factors affecting the success or failure of acquisition and adoption of net-centric systems. This effort will draw on collaboration with DIA and Lincoln Laboratories subject matter experts as well as industry and academic sources. We will use a System Dynamics analytic approach to identify and investigate the stakeholders and processes that are

involved in acquiring and sustaining IT systems along with their incentives, interactions, and dynamics. As concrete examples, we can extend our effort at looking back at systems acquired over recent years and the budgetary and other managerial policy shifts that were happening in that time frame. We will also work with DIA and Lincoln Laboratories collaborators to identify the most salient policy areas where modeling can be valuable.

(b) We plan to extend the two models described above in Sections 5 and 6 to make them: (i) more complete, (ii) more accurately match the parameters appropriate for DIA, (iii) based on (i) and (ii) perform more detailed validation studies, and (iv) to provide even deeper insights into the implications of various possible management decisions and/or situations.

(c) Also, building on the experience gained in developing the above models and additional information gathered as part of (a) above, we will develop at least one additional System Dynamics model that relate policy variables to process outcomes. This models will include various and diverse factors, both short-term and long-term, that influence programs, including budget allocations (and changes to budgets), changes requested by commanders or influential officials, contractor motivation and behavior, and other factors identified developed from the prior work and the new efforts mentioned above. It is typical that considerable insights will be uncovered in the process of creating and reviewing the resulting model.

The new System Dynamics model, as well as the improved models based on our prior work, will be exercised in simulation runs to both (a) validate and parameterize the models against observed behavior of actual scenarios, and (b) study the likely outcomes of various possible changes to improve future outcomes.

References

Berteau, David J., Guy Ben-Ari, and Matthew Zlatnik (2009). *Organizing for a Complex World*. Center for Strategic and International Studies: Washington, DC.

Corrin, Amber (2011) .” DI2E framework seeks to unite wealth of intelligence data.” Government Computer News. (May 06, 2011) http://gcn.com/Articles/2011/05/03/Defense-IT-1-intelligence-enterprise.aspx?p=1

Defense Science Board. (2009) “Department of Defense Policies and Procedures for the Acquisition of Information Technology” Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics: Washington, DC, (March 2009).

National Research Council (2010) , “Achieving Effective Acquisition of Information Technology in the Department of Defense”. Committee on Improving Processes and Policies for the Acquisition and Test of Information Technologies in the Department of Defense; National Academies Press: Washington, DC.

OSD Report to Congress (2010). “A New Approach for Delivering Information Technology Capabilities in the Department of Defense.” (Nov 2010).

PART II

Understanding the Organizational Traps in Implementing Net-Centric Systems

Xitong Li (xitongli@mit.edu)

Stuart E. Madnick (smadnick@mit.edu)

June 8, 2012

1 Introduction

In an increasingly fast-changing environment, it is important for an organization to be able to adapt its IT systems and quickly respond to changing business conditions. Such an ability is defined as the organization's information systems (IS) agility (Choi et al. 2010) and has been considered as a key facilitator to enhance dynamic capabilities (Sher and Lee 2004) and competitive advantages (Luthria and Rabhi 2009b). However, with the traditional monolithic IS architecture, it is quite difficult, expensive, and time-consuming for organizations to make changes to their IT systems (Choi et al. 2010). To address these issues, Service-Oriented Architecture (SOA) has been advocated to provide a composite set of business-aligned services that support a flexible and dynamically reconfigurable end-to-end business process realization using interface-based service descriptions (Borges et al. 2004). The net-centric strategy in the defense and intelligence communities are closely analogous to the concept of SOA and make use of SOA technology as a key element (*viz.* Croom 2006, AFEI 2008, Śliwaa and Amanowicz 2011).

One of the major objectives of adopting net-centric systems and SOA is to enhance the IS agility of organizations and improve IT-business alignment (Bieberstein et al. 2005; Choi et al. 2010; Mueller et al. 2010). Many organizations with the expectation to reap those potential benefits have invested in net-centric systems. A recent Forrester report reveals that 71% of the enterprises surveyed are already using SOA or will be by the end of 2011 (Kanaracus 2011). However, mixed outcomes about the adoption and implementation of SOA have been often reported. For example, a 2007 InformationWeek Web survey of 278 IT professionals found that 32% of those using SOA said those projects fell short of expectations and "of those, 58% said their SOA projects introduced more complexity into their IT environments, and 30% said they cost more than expected. Out of all respondents using SOAs, just 10% said the results exceeded expectations" (Smith 2008). In stark contrast, CA Technologies recently released a survey which covered 615 companies in the process of SOA-based efforts and found that 92% of their SOA initiatives met or exceeded business unit objectives (McKendrick 2011). The contradictory reported outcomes presents a paradox in the adoption and implementation of SOA and net-centric systems. The key question regarding the "implementation paradox" is why many organizations failed to meet their expectations about their implementation efforts, while others succeeded.

In the literature on net-centric systems, the potential benefits and business value of net-centric systems have been largely claimed (Cherbakov et al. 2005; Mueller et al. 2010; Varadan et al. 2008), but there are only a few empirical works actually measuring the benefits of specific implementation (Baskerville et al. 2005; Moitra and Ganesh 2005). Despite the potential benefits claimed, organizations adopting net-centric systems often encounter challenges in implementation efforts. To address the challenges, more recent (yet relatively few) literature turns to explore critical success factors (CSF) and challenges that potentially affect the implementation of net-centric systems. Research in this stream tends to enumerate a number of factors that potentially facilitate or impede organizations to receive the intended benefits of net-centric systems. However, prior research failed to explore the causal relationships between those critical factors. More importantly, none of them investigates how those critical factors interact with each other during the dynamics of the multi-month/year process of implementation of net-centric systems. Luthri et al. (2009a) extensively reviewed the literature and found that "*there is little or no realistic data available on what, if anything, firms are doing in practice to address the inherent challenges of implementing a service-oriented architecture...*" (Luthria and Rabhi 2009a). The system dynamics model

developed in this research aims to explore aspects that have not been extensively investigated in the literature.

Considering the certain technical complexity of net-centric systems and necessary organizational change, organizations adopting net-centric systems have to invest substantial resources (e.g., time and financial resources) in the implementation before they gain the potential benefits of net-centric systems. As Hau et al. (2008) argued, one of the primary challenges of SOA implementation is that "*many firms failed to realize the benefits of SOA because they suffered from the inherent tradeoff between long-term benefits versus short-term local needs of project management*" (Hau et al. 2008). From the literature on organizational theories, Repenning's works on process improvement (Repenning and Sterman 2001; Repenning and Sterman 2002) and innovation implementation (Repenning 2002) explore the challenge of capability traps in manufacturing and provide insights for us to address the challenging tradeoff faced by organizations in their implementation efforts in net-centric systems. Our research demonstrates similar but different phenomenon may happen in the failures of many implementation efforts of net-centric systems. Note that our research does not simply apply the existing theory of capability traps (Repenning and Sterman 2001; Repenning and Sterman 2002) to understanding the challenges in implementation. In fact, based on our investigation and observation, the implementation of net-centric systems has at least two inherent characteristics that are distinct from manufacturing process improvement implementation (see the discussion in the next section). Accordingly, the model and theory developed in this research suggest two different but intertwined organizational traps, i.e., technology learning trap and implementation effectiveness trap, play important roles in the failures of many implementation efforts of net-centric systems. Technology learning trap refers to the situation that the less learning in using the technology that has occurred, the more difficult and complex the technology is perceived. Implementation effectiveness trap refers to the situation in which the organization may misperceive the inappropriateness of net-centric systems when the implementation is temporally less effective and the perceived benefits of net-centric systems are delayed. Both organizational traps are distinct from the existing theory of capability traps. In addition, the second part of this research focuses on investigating how normative commitment helps to overcome the organizational traps and the learning barrier during the implementation process.

In this research, we use system dynamics modeling as the main research methodology (Sterman 2000). System dynamics has been proved to be a powerful modeling tools for organizational theory building (Repenning and Sterman 2002; Rudolph et al. 2009; Sterman 2000) and has recently received increasing attention for IS research (Choi et al. 2010; Georgantzas and Katsamakas 2008). In regard to research methodologies, Law and Urry (2004) pointed out that "*social science method has problems in understanding non-linearity relationships and flows [e.g., positive feedback loops]*" (Law and Urry 2004). Compared to other organizational research methodologies, system dynamics modeling has its unique strengths in that it allows this research to focus on the feedback loops and nonlinearity of organizational structures and time delay in the feedback loops. Besides system dynamics modeling, we use multiple research methods (including literature review, qualitative interviews, meetings/seminars and case study) for the model and theory building.

In sum, this research not only contributes to the IS literature but also sheds light on organizational theory and system dynamics literature.

The rest of this paper is organized as follows: Section 2 discusses the theoretical foundation. Section 3 presents the system dynamic model. Section 4 discusses the theory about the two organizational traps. After that, this paper turns to the second part of this research. Section 5 presents the experimental simulation results of the system dynamic model. Section 6 discusses the theory about normative commitment. Section 7 discusses the implications of this research and concludes the paper.

2 Theoretical Foundation

This research is mainly built upon three streams of the existing literature: the first two come from the literature on SOA and the third is drawn from the literature on process improvement. Each research stream provides theoretical foundation for the model and theory building.

2.1 Potential Benefits of SOA

The potential benefits and business value of SOA have been numerously claimed in the literature (Mueller et al. 2010; Varadan et al. 2008). An early work (Yoon and Carter 2007) has analyzed multiple cases and suggested that the realized benefits of SOA can be classified into two groups: improved business agility and cost reduction. The benefits contributing to improved business agility include easier integration of components and systems (Cherbakov et al. 2005), better IT-business alignment (Bieberstein et al. 2005), and a quicker response to market change or customer demand (Choi et al. 2010). The benefits of cost reduction consist of lower application development costs/time, reuse of existing components/services (Fricko 2006), and lower maintenance costs. The recent work (Mueller et al. 2010) develops a comprehensive conceptual framework to understanding the potentials of SOA. According to their work, SOA relies on three fundamental design principles: modularity, loose coupling, and standards. They built on the resource-based view and argued that SOA can enhance an organization's IS capabilities which in our research is conceptualized as IS agility (Choi et al. 2010). By enhancing IS capabilities, SOA is claimed to provide multi-dimensional benefits for organizations, including IT infrastructure, operational, strategic, managerial and organizational benefits. At the individual level, IT developers' productivity is also believed to be enhanced by improved IS reusability and interoperability from SOA design principles (Choi et al. 2010; Hau et al. 2008; Mueller et al. 2010).

Despite the numerous benefits claimed in the literature, organizations adopting SOA cannot receive those intended benefits automatically. This stream of literature generally adopts the perspective of technological determinism and fails to answer the puzzling "SOA implementation paradox": why many organizations failed to meet their expectations about SOA initiatives, while others succeeded.

2.2 Critical Success Factors of SOA Implementation

To answer the SOA implementation paradox, more recent (yet relatively few) literature turns to explore critical success factors (CSF) and challenges that potentially affect SOA implementation. The work (Luthria and Rabhi 2008) explores the organizational constraints and challenges experienced by firms considering the enterprise-wide SOA implementation. They analyzed several case studies and propose a set of seven best practices for successful enterprise-level SOA implementation. The top three are: (1) get commitment at the broad level; (2) manage expectations and invest in SOA for the long term; and (3) align the entire organization along the SOA strategy. The work (Luthria and Rabhi 2009a) presents six factors that influence the organizational adoption of SOA, among which the perceived value to the organization is ranked as the most important factor. The work (Boh and Yellin 2010) examines two organizational factors that are potentially critical in ensuring the success of SOA implementation: (1) top management support and (2) the centralization of IT decision-making. Their empirical results from hypothesis testing indicate that top management support is a significant factor, yet centralization of IT decision-making is not. The recent work (Lee et al. 2010) conducts a more comprehensive research and identifies 20 factors in SOA implementation based on their review of 34 SOA literatures and 22 interviews with both vendors and users. Their results show that "building strong support for enterprise-wide core human resources" and "clear goal-setting based on business value" are often ranked among top 3 by all the empirical data (literature review, interview with vendors and users).

In sum, the research stream on CSF tends to list a number of factors that potentially affect SOA implementation and facilitate or impede organizations to receive the intended benefits of SOA. However, prior research failed to explore the causal relationships among those critical factors. More importantly, none of them investigates how those critical factors interact with each other during the dynamics of the multi-month/year process of SOA implementation.

The abovementioned literature repeatedly points out that top management commitment and support are very critical to the success of SOA implementation. The commitment is required to be long-term and enterprise-wide, rather than short-term or local-focused. Besides that, the perceived benefits and business value from SOA are also very important to the implementation. In fact, management commitment and perceived benefits from SOA are strongly dependent; they interact with each other during the process of SOA implementation. The model and theory in our research capture this important point.

2.3 Capability Traps in Process Improvement

The third piece of theoretical foundation of this research is largely built upon Repenning's works on capability traps in process improvement (Repenning and Sterman 2001; Repenning and Sterman 2002) and innovation implementation (Repenning 2002). His research focuses on Total Quality Management (TQM) initiatives in manufacturing and develops causal-loop diagrams and system dynamic models to understand the impact of time delays between investing in process improvement and recognizing the benefits. He argues that the long delays in the feedback loops of process improvement create the dynamics of the "worse-before-better" pattern and cause capability traps and self-confirming errors. Specifically, considering the "worse-before-better" pattern, workers initially tend to underinvest in process improvement and often find themselves falling short of meeting the performance target due to insufficient process capability. Thus, workers are forced to further shift time from process improvement and increase work hours. Accordingly, the system dynamics of process improvement work as vicious cycles and workers are trapped in a downward spiral of eroding process capability, forcing less and less time for improvement. Eventually, the capability traps resulted in the failures of many process improvement efforts.

The phenomenon of beneficial improvement and innovations that go unused have been documented not only in TQM but other administrative initiatives, such as human resource practices (Pfeffer and Sutton 2000) and best practices for product development (Wheelwright and Clark 1995). The inability of many organizations to use the knowledge embodied in the improvement initiatives is a central issue facing organizational theorists (Pfeffer 1997). However, there is little IS literature documenting whether or not the similar phenomenon has happened during IS improvement efforts, especially in SOA implementation. After all, the capability traps result from the interaction between human judgmental biases and the physical structure of work processes (Repenning and Sterman 2002). This research explores whether phenomena like capability traps may also be involved in the failures of many SOA implementation efforts.

It is worth noting that this research does not simply apply the theory of capability traps to understanding the challenges in SOA implementation. In fact, based on our investigation and observation, SOA implementation has at least two inherent characteristics that are distinct from TQM and process improvement programs:

1) SOA implementation often requires enterprise-wide involvement and commitment, while process improvement programs often focus on certain work process (e.g., manufacturing, product development). It is more challenging to call for and maintain enterprise-wide, long-term involvement and commitment in organizations.
2) In regard to process improvement programs, it is relatively easier to identify defects and correct them when the process capability is low (Repenning and Sterman 2002). Thus, favorable results and positive word of mouth are easier to achieve and come earlier from the investment in improvement. Unfortunately, SOA implementation is not the case. The learning curve of the complex technology like SOA creates even longer substantial delay and postpones the potential benefits of SOA at the early stage of SOA implementation. It should be anticipated that developers may perceive little SOA effectiveness when they just start learning how to use IT systems developed by SOA design principles. Overcoming the learning-curve barrier is critical to achieve perceived benefits from SOA.

3 Model Building

3.1 Causal-Loop Diagrams

In this research, we use system dynamics modeling to develop the causal-loop diagrams for the theory building (Sterman 2000). System dynamics models consist of stocks, flows and causal links between variables. Stocks and flows are used to model physical and/or organizational processes, wherein a stock, denoted by a rectangle, represents the level that can accumulate or deplete over time. Flows denoted by straight arrows with valves cause an increase or decrease in stock levels. Stocks and flows complement feedback loops, representing the physical and/or organizational system's structure. Positive (self-reinforcing) and balancing (self-correcting) feedback loops play an important role in determining the

dynamic behaviors of organizational systems. The model consisting of causal-loop diagrams captures the key reinforcing and balancing feedback loops.

Note that the causal-loop diagrams in this research are not intended to provide an accurate mathematical specification of the relationships in the causal links; but they demonstrate the nonlinearity, discontinuities and delays between causes and effects and are valid in organizational theory building (Repenning and Sterman 2001; Repenning and Sterman 2002). Specifying a formal mathematical model is often the next step in testing the theories embodied in the causal-loop diagrams and is the second part of this paper. It is also worth noting that in this research we intentionally avoided replicating the context of a specific firm; we removed the details that were merely applied to a specific firm. The model and theory building aim to generate insights and implications in a broad organizational context.

The first assumption in our model is that only a proportion of delivered IT systems in an organization are net-centric (NC) systems. The rationale of this assumption lies in that IT developers need to spend extra time and energy to follow the net-centric principles (e.g., modularity, loose coupling, and standards) (Mueller et al. 2010) in order to make the delivered systems net-centric. Otherwise, the delivered systems just turn out to be non-net-centric and are installed in the organization. The first key variable in the model is `Proportion of NC IT Systems`, which is the ratio of the number of `NC IT Systems` to the number of total `Installed IT Systems`. Basically, `Proportion of NC IT Systems` is used to capture the penetration of net-centric systems in the organization during the implementation process. In Figure 1, `Installed IT Systems` is a stock and denoted by a rectangle.

Figure 1 shows the stock of IT system requirements backlog accumulates when system requirements are introduced over time. The delivered IT systems are installed with `System Delivery Rate`. `NC System Delivery Rate` is a fraction of the overall `System Delivery Rate`; the fraction actually depends on how much time developers spend on implementing net-centric requirements, `Fraction of Time Spent on NC REQ`. The "+" sign at the head of the causal link from `System Delivery Rate` to `NC System Delivery Rate` means there is a positive causal relationship between the two variables. That is, all other factors are equal, the higher `System Delivery Rate`, the higher `NC System Delivery Rate`. Any IT systems regardless of net-centric or not may erode over time due to the change of business environment or need for technology upgrade.

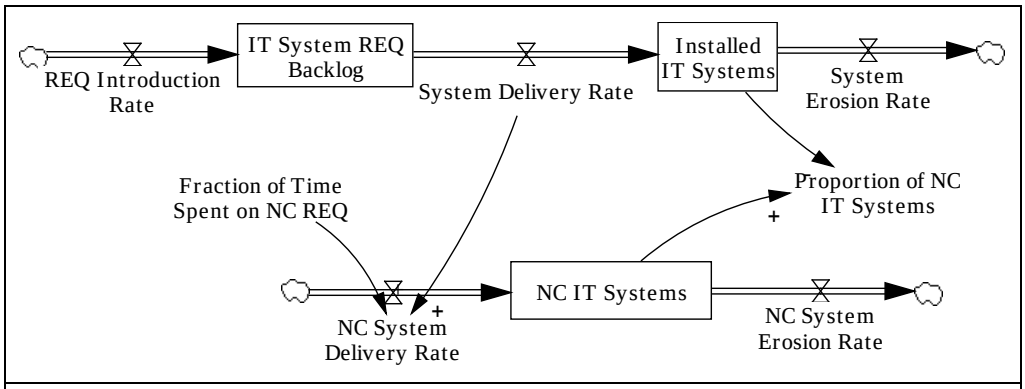


Figure 1. Proportion of Installed IT Systems are Net-Centric (NC) Systems

We present the key feedback loops in the rest of this section and then synthesize them in the causal-loop model. Readers may need to keep the entire causal-loop model (see Figure 7) in mind while reading through each of the feedback loops.

3.1.1 Balancing Loop B1: Implement Net-Centric IT Systems under Pressure

Figure 2 shows the higher `Proportion of NC IT Systems`, as the conceptualization of the organizational penetration of net-centric systems, enhance the organization's `IS agility` (Choi et al. 2010). The `IS Agility Gap`, defined as the difference between the `Desired IS Agility` and the actual `IS Agility`, results in the need and `Pressure to Implement NC Systems`. From the

perspective of most managers, Desired IS Agility is an exogenous demand. Management Commitment is also required along with IS Agility Gap to generate Pressure to Implement NC Systems, reflecting the fact that management commitment and support is a critical success factor of SOA implementation (Boh and Yellin 2010; Lee et al. 2010). Under the Pressure to Implement NC Systems, developers are forced to put a fraction of the work hours spent on implementing net-centric requirements. The time spent on net-centric requirements represents the developers need to spend extra time to follow the net-centric design principles when they develop the IT systems. The more time spent on net-centric REQ, the higher NC System Delivery Rate.

Developers not only face the Pressure to Implement NC Systems, but also the pressure to deliver IT systems on schedule, reflecting the fact that the primary tasks of developers are to develop and deliver IT systems to end users from business units. Pressure to Deliver on Schedule has two simultaneous effects on developers' work decisions: on the one hand, developers are forced to spend a large faction of time on implementing the functional requirements requested by end users. On the other hand, the developers actually shift away part of their work hours that would have been spent on net-centric requirements otherwise. In fact, when asked how developers made the tradeoff of work hours under the pressure to deliver systems on schedule, the IT manager of an interviewed organization replied:

> The requirement list we received from other departments usually put functional requirements on top of non-functional [net-centric] requirements. But those non-functional requirements were not mandatory. When we received the requirement list, we would check it and if we don't have enough time, we just cut off those non-functional requirements.... After all, we have to deliver the capabilities [functionalities of the IT systems] to our end users within the limited schedule and resources.

Therefore, under the two kinds of pressures, developers first decide how much time they would spend on the functional development and then spend the rest of their time on net-centric requirements. In Figure 2, the balancing loop B1 represents the fact that developers implement net-centric IT systems under the two kinds of pressures. By a balancing loop B1, it suggests that the IS Agility Gap is being closed over time when more net-centric systems are implemented and installed, releasing the Pressure to Implement NC Systems.

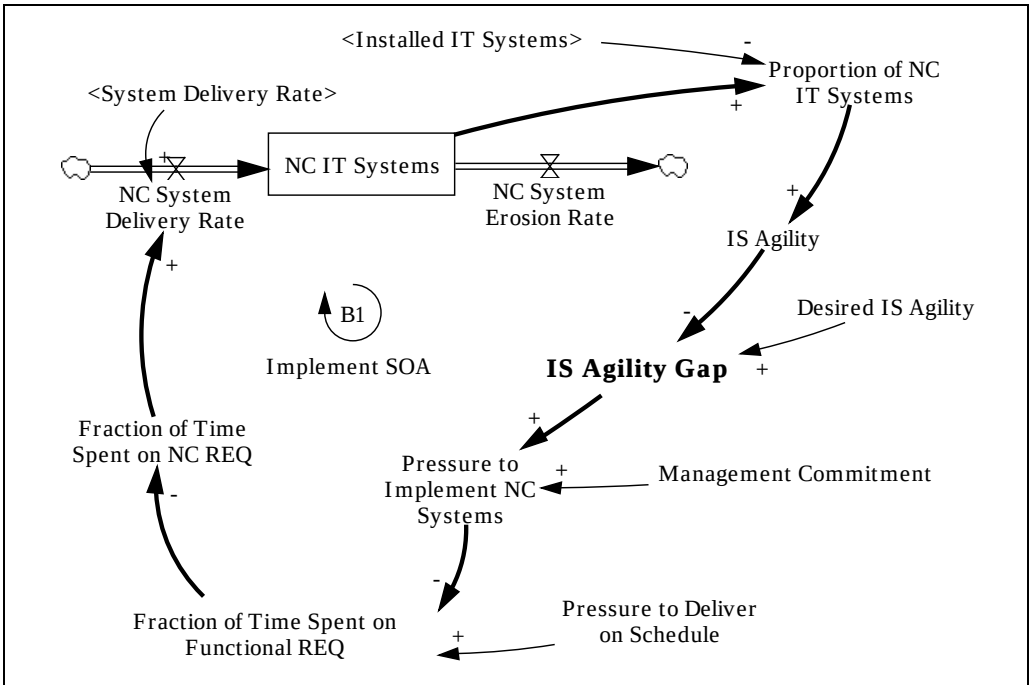


Figure 2. Balancing Loop B1: Implement Net-Centric IT Systems under Pressure

3.1.2 Balancing Loop B2: Work Harder to Deliver on Schedule and Bypass NC REQ

Figure 3 shows the normal work structure of IT developers in their daily work lives. With the `IT System REQ Backlog` and the cycle time requested by business units for delivering IT systems, IT managers and developers calculate the `Desired Delivery Rate`. The IT department has its actual `System Delivery Rate` which is determined by `Developer Headcount`, a developer's average `Development Productivity`, and how much time developers need to spend on functional requirements of the IT systems. Note that `Development Productivity` and `System Delivery Rate` are two distinct performance indicators. `Development Productivity` refers to on average how many IT systems that a developer can deliver within one unit of time (say one month) when the developer spends all of his work hours on the development of functionalities. Differently, `System Delivery Rate` refers to how many IT systems that the development team as a whole (e.g., the entire IT department) can deliver within one unit of time during which the developers may spend part of their work hours on implementing net-centric requirement or attending training sessions about net-centric systems, etc. From the perspective of managers, `System Delivery Rate` is an aggregate-level and more salient performance indicator. A manager who used to be the CIO of a large US university told us that:

> As a manager, I usually don't care much about a single developer's productivity. I always care about how fast we [as a team] are able to deliver the systems requested [by business units]. In other words, we care the system delivery rate much.

`Delivery Rate Gap` refers to the difference between `Desired Delivery Rate` and `System Delivery Rate`. Unlike `IS Agility Gap` creating `Pressure to Implement NC Systems`, the `Delivery Rate Gap` can create the `Pressure to Deliver on Schedule`. In case of high pressure to deliver on schedule, developers have to work harder on functional development and try to catch up the delivery schedule. Thus, developers are often forced to spend a large faction of time on implementing the functional requirements and bypass net-centric requirements. The balancing loop B2 captures such a situation in which developers tend to bypass net-centric requirements and work harder to get their development jobs done under the schedule pressure. Balancing loops B2 works to close the `Delivery Rate Gap` and release the `Pressure to Deliver on Schedule`.

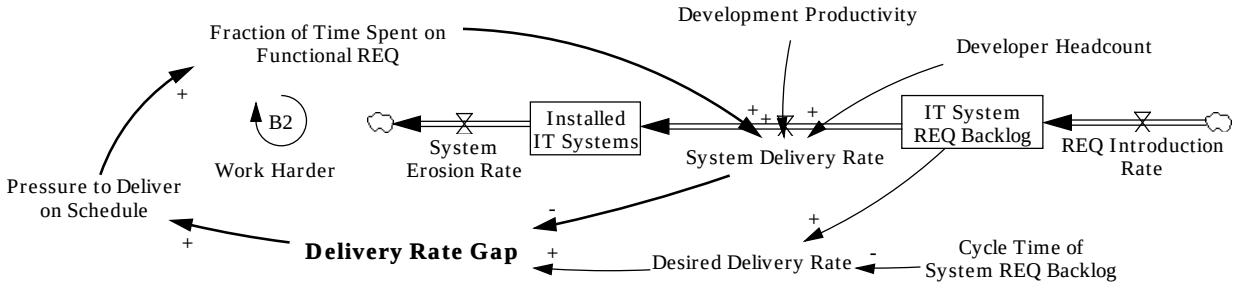


Figure 3. Balancing Loop B2: Work Harder to Deliver on Schedule and Bypass NC REQ

3.1.3 Reinforcing Loop R1: Implement NC Systems through Productivity Increase

Now we turn to the potential benefits of implementation of net-centric systems. Net-centric systems, developed using net-centric design principles (e.g., modularity, loose coupling and standards), are more reusable, interoperable and easier to integrate with IT systems (Choi et al. 2010; Mueller et al. 2010). Therefore, it is easy for developers to make use of the existing reusable net-centric systems (e.g., components or services) when they develop new IT systems and integrate them with existing service-oriented systems that are already installed in the organization. In this sense, net-centric implementation actually makes the development job easier and allows developers to develop IT systems faster. In other words, net-centric systems increase `Development Productivity` of the developers on average (Choi et al. 2010; Hau et al. 2008; Mueller et al. 2010). In Figure 4, we use a key variable `Effectiveness of NC Systems` to represent the extent to which the developers' `Development Productivity` is

increased, on average, compared to their Base Development Productivity in the basic situation prior to net-centric implementation. Specifically, the more net-centric systems installed in the organization (i.e., the higher Proportion of NC IT Systems), the more Effectiveness of NC Systems the developers enjoy and in turn the higher their average Development Productivity. The rise of Development Productivity results in the increase of System Delivery Rate.

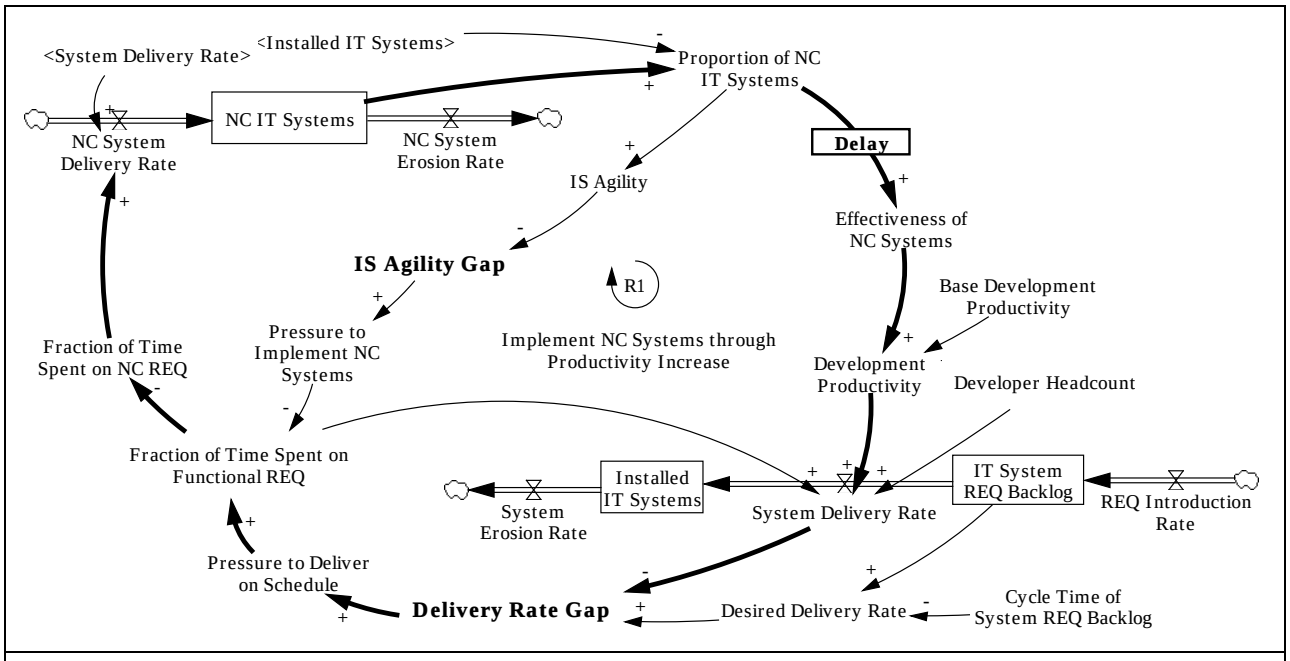


Figure 4. Reinforcing Loop R1: Implement NC Systems through Productivity Increase

With higher System Delivery Rate, Delivery Rate Gap is closed and the Pressure to Deliver on Schedule is released. As a result, developers with less schedule pressure are more likely to spend more time on developing net-centric IT systems, increasing the NC System Delivery Rate and the number of installed NC IT Systems. Eventually, the Proportion of NC IT Systems will rise further and net-centric systems become more effective, which further enhance the developers' Development Productivity. The entire process becomes a reinforcing loop R1 which is labeled as "Implement NC Systems through Productivity Increase", as shown in Figure 4.

The reinforcing loop R1 can operate as either virtuous or vicious cycles, depending on its current state (Sterman 2000). When R1 operates as virtuous cycles, more net-centric IT systems are implemented (i.e., more penetration of net-centricity in the organization), net-centric systems become more effective and allow developers to invest more time on further implementation. Conversely, when R1 operates as vicious cycles, less net-centric penetration in the organization generates little effectiveness of net-centric systems and contributes little to the developers' development productivity. Under the schedule pressure, developers are likely to shift more time which would have been spent on net-centric implementation to the functional development. Consequently, less service-oriented systems are developed and the net-centric penetration becomes even less.

In order to successfully implement net-centric systems, the reinforcing loop R1 operating as virtuous cycles is preferable. However, this cannot always be the case. Many organizations struggling with their net-centric implementation did actually suffer from the vicious cycles of the dynamics (Hau et al. 2008).

It is worth noting that there is a substantial delay between Proportion of NC IT Systems and Effectiveness of NC Systems, because it takes time for developers to attend training sessions and introduce new work structure so as to acquire sufficient knowledge about how to make use of existing reusable components/services and implement new service-oriented systems. Choi et al. (2010) explicitly documented that "the *learning curve and the introduction of the governance mechanism will introduce a*

delay in implementation, as opposed to using current technology". We will discuss the impact of the delays in the next section.

3.1.4 Reinforcing Loop R2: Motivate Commitment through Perceived Benefits of NC Systems

As discussed, `Effectiveness of NC Systems` increases the developers' `Development Productivity`, compared to their `Base Development Productivity`. Therefore, the benefits and value of net-centricity will be perceived by the organizational participants (e.g., managers and developers). The positive relationship between the effectiveness of innovative IT (particularly the effectiveness of net-centric systems) and the perceived benefits has been largely discussed (Mueller et al. 2010). In particular, Choi et al. (2010) pointed out that net-centric implementation effectiveness is an important determinant of the perceived benefits and value derived from net-centric systems.

Greater perceived benefits of net-centric systems create favorable word of mouth in the organization (Sterman 2000) and generate the additional commitment to net-centric implementation internally. The causal link between results of the technology in use and the commitment generation has been supported by many motivation and organizational theories (Repenning 2002; Vroom 1964). In this research, the commitment generated by perceived benefits of net-centric systems are considered to be internal or endogenous, emphasizing the additional commitment actually results from the results attribute to the use of net-centric IT systems. This is also supported by social cognition theory saying "performance successes strengthen self-beliefs of capability" (Wood and Bandura 1989).

Besides the endogenous sources of commitment, there are exogenous sources of commitment which are labeled as `Normative Commitment`. Institutional theory suggests that coercive, mimetic and normative pressures are important factors affecting the innovation adoption (DiMaggio and Powell 1983). The work (Liang et al. 2007) on the assimilation of enterprise systems also provides support that institutional pressures positively affect top management participation in the ERP assimilation process. In other words, management commitment and participation mediate the effects of institutional pressures on IT assimilation (Liang et al. 2007).

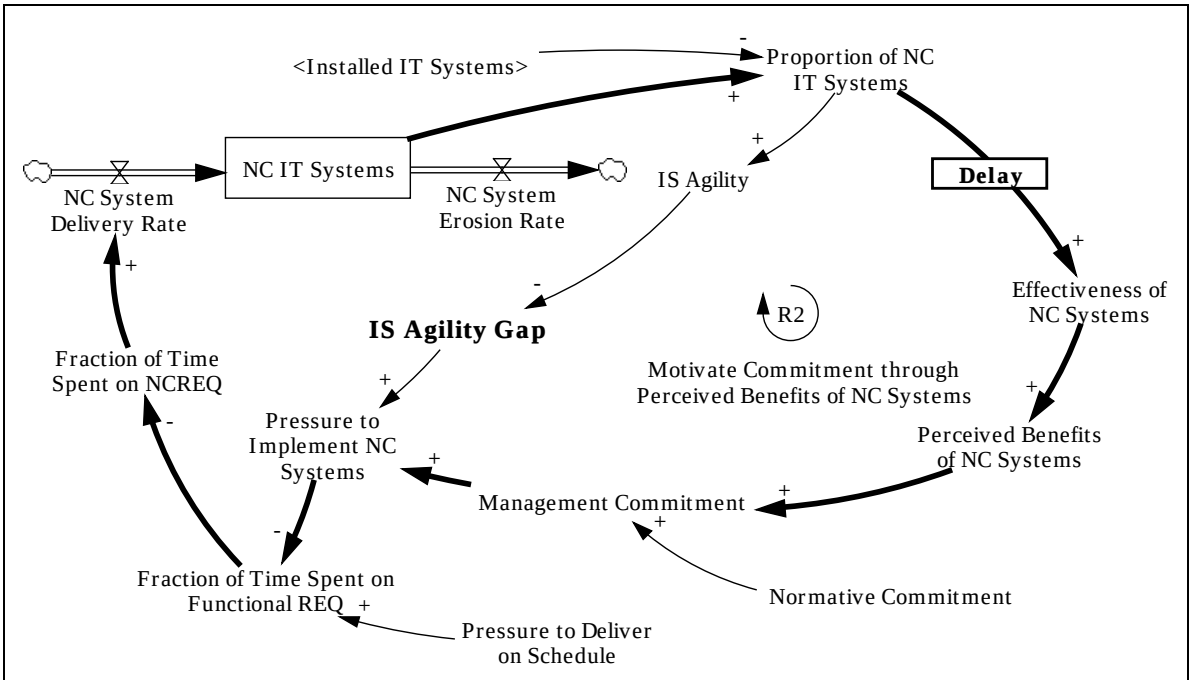


Figure 5. Reinforcing Loop R2: Motivate Commitment through Perceived Benefits of NC Systems

Figure 5 shows the mediating role played by Management Commitment between Normative Commitment and net-centric implementation. Management Commitment generates Pressure to Implement NC Systems and forces developers to spend part of their work hours on implementing net-centric systems, even though they are still under the Pressure to Deliver on Schedule. When net-centric IT systems are implemented and installed over time, net-centric implementation becomes more effective and enhances developers' Development Productivity. With more benefits of net-centric systems perceived by the organization, the effects of word of mouth would generate internal commitment endogenously and in turn enhance the total Management Commitment. The reinforcing loop R2, labeled as "Motivate Commitment through Perceived Benefits of NC Systems", represents the dynamics of the commitment motivation. Similar to R1, the reinforcing loop R2 operating as virtuous cycles is clearly preferable and highlights the importance of management commitment, which is consistent with the existing empirical evidence (Boh and Yellin 2010; Lee et al. 2010). Likewise, there is substantial delay in the reinforcing loop R2.

3.1.5 Reinforcing Loop R3: Acquiring Knowledge of Net-Centricity by Doing

The time delay between Proportion of NC IT Systems and Effectiveness of NC Systems plays an important role in the dynamics of the reinforcing loops R1 and R2. Now we need to explore from where the time delay comes. According to our conversations with managers and developers from the organizations implementing net-centric systems, the Effectiveness of NC Systems depends on two factors. The one factor is Proportion of NC IT Systems, which means net-centricity is more effective with more net-centric IT systems installed in the organization. In other words, when more installed IT systems are service-oriented (e.g., loose-coupling, interoperable), it becomes easier to integrate with those installed IT systems when developers build new IT systems. The other factor is Knowledge of Net-Centricity, the organization's knowledge about how to use and build net-centric IT systems. Without sufficient Knowledge of Net-Centricity, developers may still feel difficult to make use of existing net-centric IT systems or build new IT systems using net-centric design principles, even if Proportion of NC IT Systems is high. Knowledge of Net-Centricity is acquired over time when developers persist to make use of existing net-centric IT systems and adhere to net-centric design principles. In fact, net-centric systems, as a new architectural style (Borges et al. 2004), have certain technical complexity and its implementation process is characterized by a high learning barrier (Choi et al. 2010). In order to realize the effectiveness of net-centric systems, developers have to invest substantial time and energy in learning the new architectural style, e.g., learning net-centric design principles and methodologies (Mueller et al. 2010).

We use the first-order information delay to model the process through which Knowledge of Net-Centricity is acquired, as shown in Figure 6. A key construct Learning Time is used to capture the time delay of the learning process. Learning Time also represents the technical complexity of net-centric systems to the organization. That is, if net-centric systems are more technical complex, the organizational learning time is expected to be longer. Note that Learning Time depends not only on the technical complexity itself, but also on organizational contexts, such as organization size. We expect the Learning Time is longer for a larger organization than that for a small organization.

In Figure 6, there are two key factors that affect the Learning Rate of Knowledge of NC: one is Proportion of NC IT Systems and the other is Perceived Benefits of NC Systems, both of which would increase the learning rate. When more service-oriented IT systems are installed in the organization (i.e., higher Proportion of NC IT Systems), it is easier to use and learn net-centric IT systems. Also, if net-centric systems are perceived to be more beneficial, developers are more willing to adhere to the net-centric design principles in their development work and thus they would acquire knowledge of net-centricity faster.

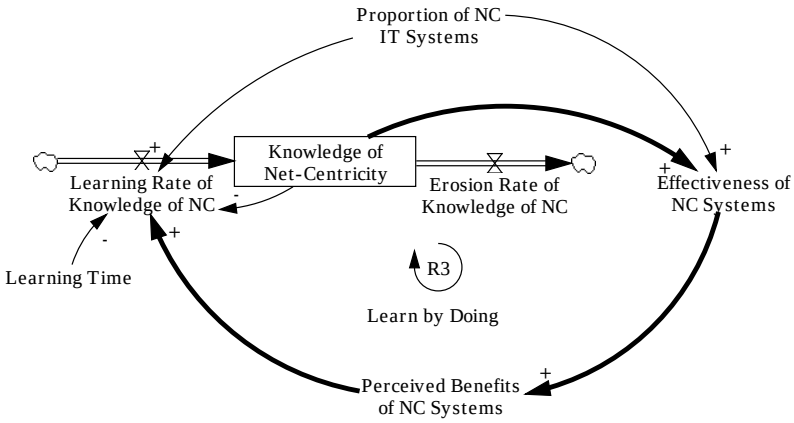


Figure 6. Reinforcing Loop R3: Acquiring Knowledge of Net-Centricity by Doing

As shown in Figure 6, when the `Effectiveness of NC Systems` is realized and more benefits of net-centric systems are perceived by the developers, they are more willing to adhere to net-centric design principles and learn how to use and build net-centric IT systems. The adherence to net-centricity will increase the `Learning Rate of Knowledge of NC` and the organizational `Knowledge of Net-Centricity`, which in turn makes net-centric systems more effective in the organization. The reinforcing loop R3 represents the process through which developers acquire knowledge of net-centricity by doing.

3.2 The Model

We synthesize the key feedback loops in the system dynamic model, as shown in Figure 7.

The balancing loop B1 indicates developers invest part of their work hours to implement net-centric IT systems. There are two kinds of pressures that affect the tradeoff decision of IT developers and managers: one is the pressure to deliver the functionalities of IT systems on schedule, and the other is the pressure to implementation of net-centric systems. The balancing loop B1 operates to close the organization's IS agility gap over time, yet management commitment to net-centricity plays the important role in generating the momentum for implementing net-centric systems.

The balancing loop B2 represents the decision that developers would naturally make to work harder and to get the functional development done on schedule. The decision actually shifts part of their work hours to the functional development and under-invests in net-centricity. B2 indicates that the tendency of developers that puts high priority on delivering functionalities to end users on schedule, which is confirmed by many managers that we interviewed. B2 operates to close the delivery rate gap and release the schedule pressure. It is worth noting that there is relatively shorter delay within B2 than the delay within B1. In case of a big delivery rate gap (e.g., urgent IT functionalities are requested by end users from business units), it is very likely for the IT managers and developers to make the decision that bypasses the net-centric requirements and accelerates the development of functionalities. This is because `System Delivery Rate` is a more salient performance indicator associated to closing the `Delivery Rate Gap` more quickly, while `IS Agility` is an organizational, less salient performance indicator. That human tend to overemphasize salient factors when processing attributions is a well-known cognitive and perceptual bias (Tversky and Kahneman 1974). Since bypassing net-centric requirements only hurts the IS agility in the long term, it is difficult for people to attribute to such shortcuts after the substantial delay. But closing the `Delivery Rate Gap` more quickly may probably bring IT managers and developers favorable gains or avoid negative words from other organizational participants (e.g., end users from business units).

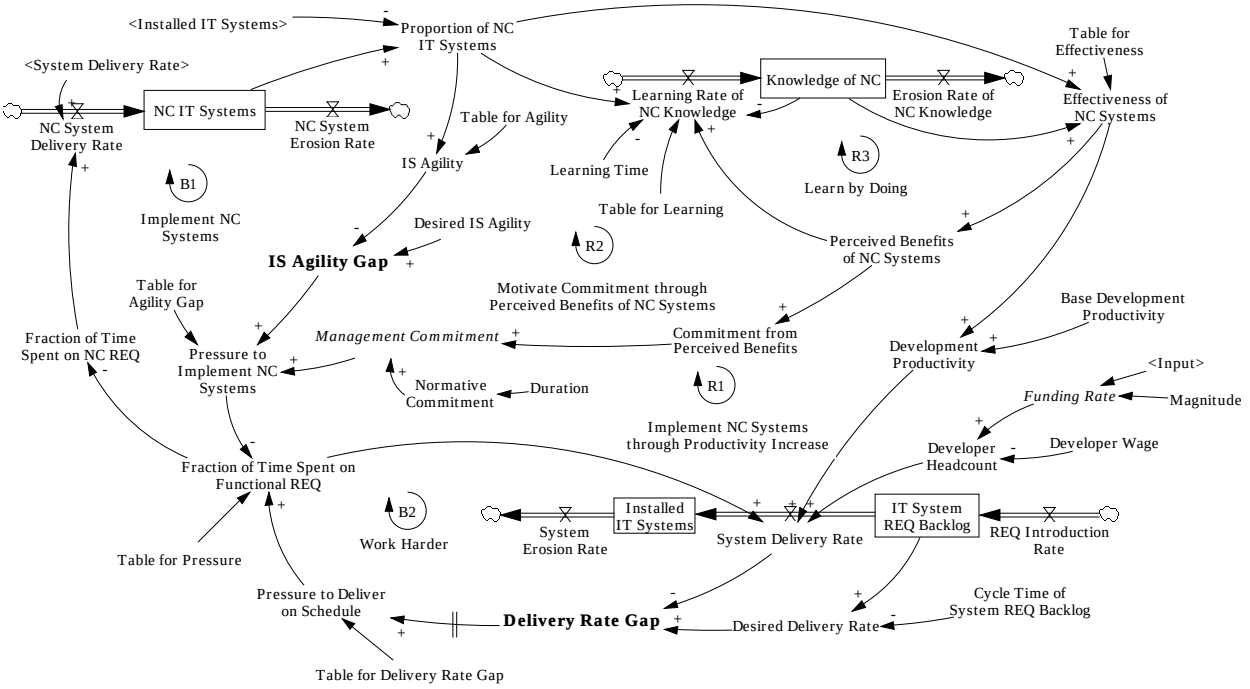


Figure 7. The Model of Implementing Net-Centric Systems

The reinforcing loop R1 represents developers' development productivity increase over time with more and more net-centric systems installed in the organization, releasing the `Pressure to Deliver on Schedule`. Under less intense schedule pressure, it is more likely for IT managers and developers to invest their work hours in implementation of net-centric systems. However, developers cannot immediately acquire the knowledge of net-centricity considering the technical complexity (Choi et al. 2010) and thus cannot increase their development productivity in a short period. The substantial delay in the reinforcing loop R1 has two important effects on the decisions of IT managers and developers. On the one hand, the substantial delay suggests managers have to keep investing in net-centric systems for a long time and sacrifice the system delivery rate before the development productivity takes off. This means the "worse" period of the "worse-before-better" pattern (Repenning and Sterman 2001) may last long, but apparently, not every organization or manager would tolerate a much long "worse" period. On the other hand, the substantial delay makes the causal link between the effectiveness of net-centric systems and system delivery rate uncertain and less salient. It is thus difficult for people to attribute the rise of system delivery rate to the investment in implementing net-centric systems several months or even years ago, because people tend to attribute to more available and salient causes due to cognitive biases (Tversky and Kahneman 1974).

The reinforcing loop R2 represents an organization perceives more benefits from net-centric implementation when the development productivity is being improved over time. The perceived benefits of net-centric systems motivate the management commitment from the internal environment of the organization and promote the net-centric implementation further, resulting in more perceived benefits. Similar to R1, R2 is also subject to the substantial delay between higher net-centric penetration and the perceived benefits of net-centric systems. Yet the substantial delay in R2 has a very important effect that is different from the two effects of the delay in R1. That is, the substantial delay in R2 largely postpones the potential benefits of net-centric systems to be perceived by the organization. Accordingly, organizations that decide to implement net-centric systems have to be patient enough and tolerate a perhaps long period during which little benefits of net-centric systems are perceived, especially at the early stage of implementation of net-centric systems. Thus, the primary part of management commitment to net-centricity has to come from normative commitment. In such situations, top management (e.g.,

CIOs) has to use their leadership to resist possibly unfavorable initial word of mouth about net-centricity. The normative commitment has to be maintained for long enough before perceived benefits of net-centric systems arrive and enough endogenous commitment is motivated. The implication is that top management's leadership in maintaining normative commitment is important to leverage the benefits of IT in general (Armstrong and Sambamurthy 1999) and net-centric systems in particular.

The reinforcing loop R3 models the mechanism of the time delay between `Proportion of NC IT Systems` and `Effectiveness of NC Systems`. The time delay is subject to the organizational learning process of knowledge of net-centricity. It takes time for an organization to acquire knowledge of net-centricity through using and building net-centric IT systems and adherence to net-centric design principles.

4 Theory on Organizational Traps

In this section, the theory about organizational traps that impede the net-centric implementation is built before the simulation experiments.

4.1 Organizational Traps

There is substantial delay from the investment in net-centricity to the rise of development productivity and the perceived benefits of net-centric systems. Considering the reinforcing loops R1 and R2, we postulate the learning curve of net-centricity is likely to be very low and flat at the early (but probably long) stage of the implementation and to take off at some point and grow up quickly. As Hau et al. (2008) observed, it is quite common that "*the first release of an SOA application to take additional time because adherence to SOA design principles often leads to longer design time without yielding immediate benefits*" (Choi et al. 2010; Hau et al. 2008). Due to the substantial delay, managers and developers are likely to bypass net-centric requirements and/or underinvest in net-centric systems, especially when urgent IT functionalities are requested by end users from business units and intense schedule pressure is created. A project manager that we interviewed told that:

> There are actually "waiver processes". When the requests [of IT functionalities] from end users are urgent enough or some emergence happens, they can apply for the waiver and don't need to go through the whole process [e.g., bypassing net-centric requirements]. This is a tactic vs. strategic balance in our organization. And different departments actually have different waiver processes...

Underinvestment in net-centricity postpones the first release of a net-centric application and perhaps allows delivering non-service-oriented systems, leading to less penetration of net-centricity in the organization and in turn less effectiveness of net-centric implementation. As a result, the perceived benefits of net-centric systems are further delayed and negative word of mouth may spread across the organization. For example, the technical complexity of net-centric systems is overemphasized, instability of open standards for net-centricity and inappropriateness of net-centricity are misperceived (Choi et al. 2010; Hau et al. 2008).

In such situations, the organization is likely to be stuck in two different traps: technology learning trap and implementation effectiveness trap.

Technology learning trap refers to the situation that the less learning in using the technology, the more difficult and complex the technology is perceived. Technology learning trap indicates a vicious cycle of the "learning-by-doing" (Arrow 1962) or more specifically "learning-by-using" process (Rosenberg 1982). Consequently, developers may continue to underinvest in net-centric systems and thus the low, flat part of the learning curve of net-centricity is actually prolonged, which postpones the perceived benefits of net-centric systems even further. Thus, the organization is trapped in the low, flat and prolonging part of the learning curve of net-centricity and can hardly see it taking off.

Implementation effectiveness trap refers to the situation in which the organization may misperceive the inappropriateness of net-centric systems when net-centric implementation is temporally less effective and perceived benefits of net-centric systems are delayed. It is likely for the organization to falsely conclude that net-centricity is inappropriate to its organizational context, rather than to admit that it is just due to insufficient net-centric penetration in the organization. As a result, developers continue to underinvest in net-centric systems and deliver non-net-centric systems, further undermining the

effectiveness of net-centric implementation. In such a case, the organization is stuck in the trap and can hardly realize the effectiveness of net-centric systems.

Technology learning trap and implementation effectiveness trap are different but intertwined with each other. Because top management attention (and resources) often fades away over time (Kotter 1995), the two traps result in the failures of net-centric implementation efforts of many organizations when the normative commitment of top management to net-centricity fades away.

5 Experimental Simulation

In this and next section, we turn to investigate how to overcome the technology learning trap and the implementation effectiveness trap. Specifically, we focus on exploring how the duration of the normative commitment affects the dynamics of the implementation process. We use experimental simulation to address this question.

In order to operationalize the system dynamics model (see Figure 7), we used 5 table functions in the model. We performed several kinds of model validation to enhance the validity of the model, including model check and unit check. Our model has successfully passed the model validation.

5.1 Base Simulation

In the experimental simulation, we currently focus on two key variables: one is the duration of the normative commitment, and the other is the learning time. In the base model, the duration of the commitment is set to be 12 months and the learning time is 12 months also. This means that the normative commitment lasts for 12 months and after that, top management simply removes the normative commitment. We also assume that a developer becomes more knowledgeable and effective about net-centric methods over time, with 12 months being the average time to be fully competent.

Figure 8 shows the simulation results of the base model. As shown in Figure 8(a), the organization's IS agility grows gradually before the first 12 months. When the normative commitment is removed at the 12th month, the IS agility turns to drop for a while. But at around the 15th month, the IS agility stops to drop and turns to grow again. Eventually, the organization's IS agility grows to equilibrate at a very high level. Thus, this result reveals that it is a good case in which the organization has successfully built up its IS agility through implementing net-centric systems.

As shown in Figure 8(b), most of the developers' work hours are spent on functional requirements when the normative commitment is removed suddenly at the 12th month. But since there may be sufficient internal commitment already motivated through perceived benefits of net-centric systems, the developers would like to spend a certain fraction of their time on implementing net-centric requirements. With more benefits of net-centric systems are perceived, the developers would like to spend more time on net-centric requirements and thus less time on functional requirements. Figure 8(b) shows the fraction of time spent on functional requirements drops gradually after the sudden rise at the 12th month. When the organization builds up sufficient IS agility, developers, with less pressure to implement net-centric systems, can turn to spend more time on their primary jobs, i.e., functional requirements.

Figure 8(c) indicates that when the normative commitment is removed at the 12th month and developers bypass net-centric requirements, the entire system delivery rate actually rises suddenly. But in order to implement net-centric systems and build up the IS agility, the organization does suffer from the "worse-before-before" phenomenon, that is, the system delivery rate drops during the 12th to 20th month. Only after the 20th month, the system delivery rate turns to grow gradually and exceeds the highest point at the 12th month and equilibrates at a higher level. Figure 8(d) shows that the IT system requirement backlog also equilibrates at a certain level.

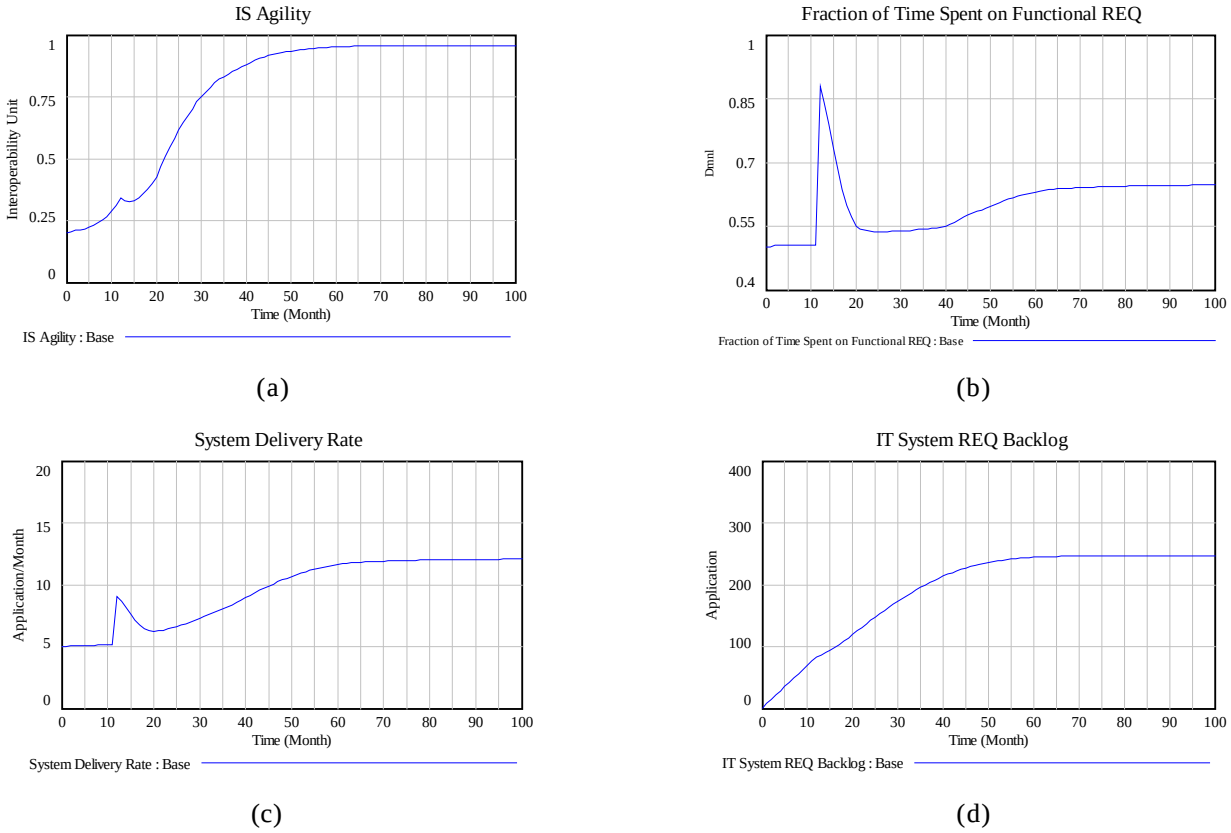
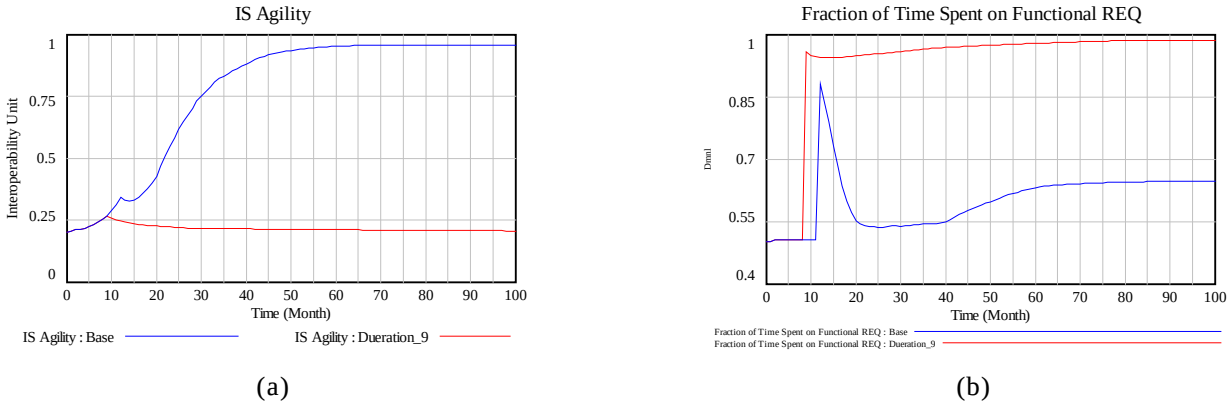


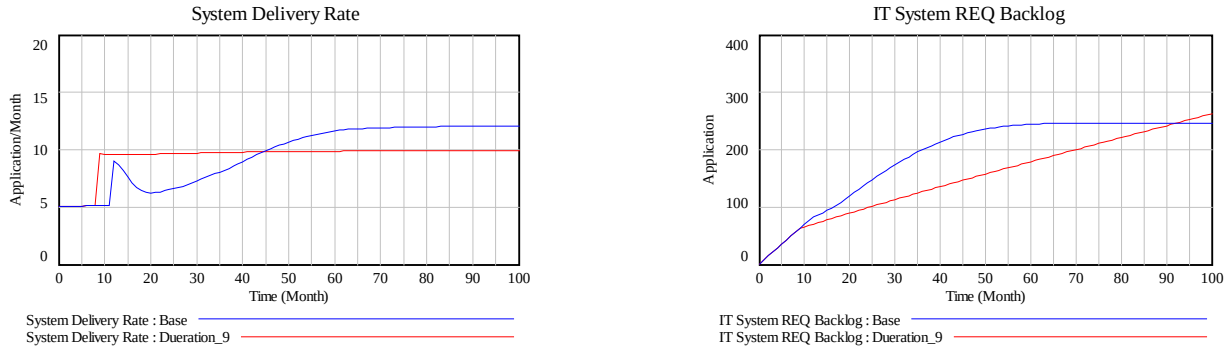
Figure 8. Simulation Results of the Base Model

In sum, the simulation results of the base model demonstrate a case in which the organization removing its normative commitment at the 12th month can still successfully implement net-centric systems and build up the IS agility. Removing normative commitment and bypassing net-centric requirement do increase the system delivery rate. But in order to successfully implement net-centric systems, the organization does suffer from a "worse-before-better" phenomenon.

5.2 Simulation of Experiment One

In the first experimental simulation, we change the duration of normative commitment from 12 months to 9 months, holding other factors unchanged. That is, we assume the normative commitment is removed at the 9th month, three months before the base simulation. This experimental simulation allows us to focus on examining how the shorter duration of normative commitment affects the dynamics of the implementation process.





(c) (d)

Figure 9. Simulation Results of Experiment One

Figure 9 shows the simulation results of Experiment One, comparing to that of the base simulation. As shown in Figure 9(a), the organization's IS agility turns to decrease after the normative commitment is removed at the 9th month and can never grow up. Eventually, the IS agility stays at a low level just as the level from the very beginning. This result reveals that little effectiveness of net-centric systems has been realized when the normative commitment is removed at the 9th month. Without realizing the effectiveness of net-centric systems, little benefit of net-centricity is perceived in the organization, resulting in little internal commitment on net-centricity. In this case, the organization simply stops implementing net-centric systems. As shown in Figure 9(b), the developers spend most of their work hours on functional requirements and thus little time on net-centric requirements.

Figure 9(c) indicates that the system delivery rate jumps to a higher level when the normative commitment is removed at the 9th month and developers spend little time on net-centric requirements. But the system delivery rate remains at that level which is actually lower than the case of the base simulation. This means the organization actually loses the opportunity of implementing net-centric systems to increase its system delivery rate. Also, the organization loses the opportunity of implementing net-centric systems to enhance its IS agility, as shown in Figure 9(a).

Figure 9(d) shows that the IT system requirement backlog always grows up, which means the IT department cannot complete the system requirements requested by the business units. In contrast, the IT department can accomplish the system requirements in the case of base simulation.

5.3 Simulation of Experiment Two

In the second experimental simulation, we change the learning time from 12 months to 16 months, holding the duration of normative commitment as 12 months and other factors unchanged. That is, we assume the learning time for a certain organization to acquire knowledge of net-centricity is 16 months, rather than 12 months, and the normative commitment is removed at the 12th month as the base simulation. This experimental simulation allows us to focus on examining how the longer learning time, in terms of a given duration of normative commitment, affects the dynamics of the implementation process.

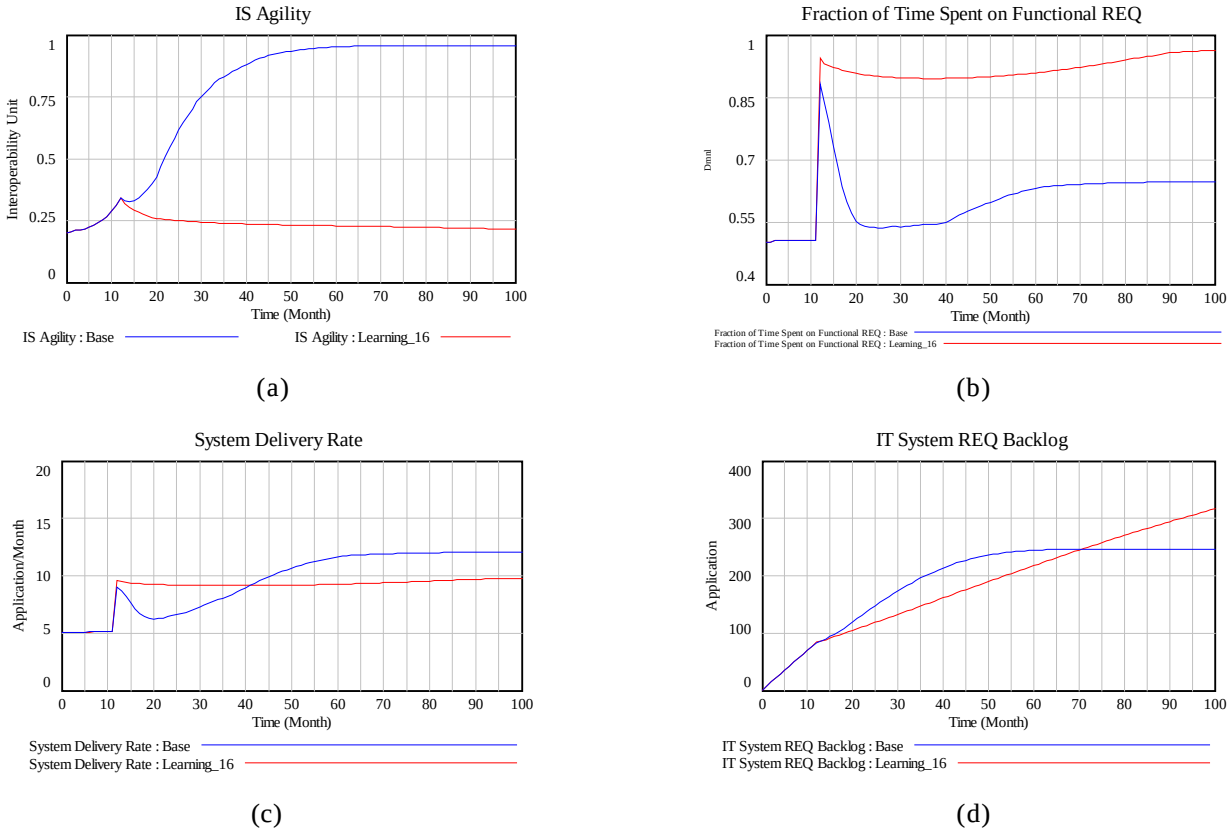


Figure 10. Simulation Results of Experiment Two

Figure 10 shows the simulation results of Experiment Two, comparing to that of the base simulation. As shown in Figure 10(a), the organization's IS agility turns to decrease after the normative commitment is removed at the 12th month and can never grow up. Eventually, the IS agility gradually falls to the low level just as that from the very beginning. This result reveals that little effectiveness of net-centric systems has been realized when the normative commitment is removed at the 12th month. Without realizing the effectiveness of net-centric systems, little benefit of net-centric systems is perceived in the organization, resulting in little internal commitment on net-centricity. In this case, the organization simply stops implementing net-centric systems. As shown in Figure 10(b), the developers spend most of their work hours on functional requirements and thus little time on net-centric requirements.

Figure 10(c) indicates that the system delivery rate jumps to a higher level when the normative commitment is removed at the 12th month and developers spend little time on net-centric requirements. But the system delivery rate remains at that level which is actually lower than the case of the base simulation. This means the organization actually loses the opportunity of implementing net-centric systems to increase its system delivery rate. Also, the organization loses the opportunity of implementing net-centric systems to enhance its IS agility, as shown in Figure 10(a). Figure 10(d) shows that the IT system requirement backlog always grows up, which means the IT department cannot complete the system requirements requested by the business units. In contrast, the IT department can accomplish the system requirements in the case of base simulation.

6 Theory on Normative Commitment

According to the experimental simulation, we can see that the dynamics of Experiment One and Experiment Two are very similar, in both of which the organization fails in the net-centric implementation and loses the opportunity to enhance its IS agility.

Experiment One tells that given the learning time 12 months, lasting the normative commitment for 12 months is enough to succeed in the net-centric implementation, while lasting the normative commitment

for 9 months is not. If the normative commitment is removed at the 9th month (only 3 months early), the organization does not realize much effectiveness of net-centric systems and thus perceives little benefits of net-centric systems. With little perceived benefits of net-centric systems, the organization participants (developers and managers) are not convinced that net-centricity is appropriate to their organization. Negative word of mouth about net-centric systems will be generated and spread in the organization, resulting in less pressure to implement net-centric systems. On the other hand, developers' development productivity is not improved and developers still face large pressure to deliver the functional requirements on schedule. Consequently, developers will shift the time spent on net-centric requirements to the development of functional requirements and develop non-net-centric IT systems in the organization. This results in less proportion of net-centric IT systems and even less effectiveness of net-centric systems. In addition, developers do not adhere to net-centric design principles due to less perceived benefits of net-centric systems, lowering the learning rate of knowledge of net-centricity. In this case, all the three reinforcing loops R1~R3 operate as the vicious cycles and the net-centric implementation fails.

Experiment Two tells that it is not enough for an organization with 16-month learning time to last the normative commitment for 12 months, while it is enough for an organization with 12-month learning time. If the learning time of an organization is 16 months rather than 12 months (for example, a larger organization or long-standing may have longer learning time, compared to a small or new organization), the organization needs to put longer normative commitment. In other words, the minimum duration of the normative commitment is probably positively associated with the learning time of the organization to acquire knowledge of net-centricity. The organization has to last the normative commitment long enough in order to acquire sufficient knowledge of net-centricity and realize sufficient effectiveness of net-centric systems. Otherwise, if the normative commitment fades away at the midway, the three reinforcing loops R1~R3 operate as the vicious cycles and the organization is likely to fail in the net-centric implementation.

Therefore, the theory herein is about the relationship between normative commitment and organization learning time. We summarize the theory in the following proposition:

Proposition: *Certis parabis, organizations with longer learning time need to last the normative commitment longer in order to acquire sufficient knowledge of net-centricity and realize the effectiveness of net-centric systems, succeeding in the net-centric implementation.*

7 Discussion

In this research, we assume to a large extent that net-centric systems, once implemented well in an organization, can enhance the organization's IS agility and developers' development productivity at least in the long term (Choi et al. 2010; Hau et al. 2008; Mueller et al. 2010; Yoon and Carter 2007). Despite best practices of net-centric implementation offered by vendors, consultants and leading organizations (Krafzig et al. 2005), many organizations encountered organizational and human challenges in their net-centric implementation efforts (Fricko 2006; Luthria and Rabhi 2008). In fact, prior research suggests that human agency often plays an important, and sometimes critical, role in organization-wide IS implementation, like Enterprise Resource Planning (ERP) implementation (Orlikowski 1992; Volkoff et al. 2007). For example, Grant et al. (2006) write "*key stakeholders in the ERP implementation process adopted different discourses*" and highlighted the role of their discourses in the social shaping of ERP implementation (Grant et al. 2006). Scott el al. (2003) further point out that the "success" or "failure" of the ERP implementation is actually highly situated and relate to "*the negotiations between actor networks surrounding the implementation process*" (Scott and Wagner 2003). Although ERP as a monolithic IS architecture is very different from net-centric systems and thus has different organizational implications, prior research on ERP implementation leads us to accommodate human agency in this research and to examine the important role that human agency plays during the implementation process of net-centric systems.

Net-centric implementation, like any organization-wide IS implementation, requires organizations to invest substantial resources upfront before potential benefits are perceived by organizational participants, known as the "worse-before-better" phenomenon. That is, there are substantial delays between the implementation investment and the perceived benefits. During the "worse" period of the implementation, different organizational participants often make different senses and judgments about what becomes

"worse" to them, how "worse" it will be, and how long the "worse" will last. Impatient organizational participants are likely to underinvest in net-centric implementation and thus the dynamics can get stuck in two intertwined traps: technology learning trap and implementation effectiveness trap. Once stuck in the traps, it would be difficult for the organizational participants to correctly attribute to the vicious cycle of the dynamics of the implementation process. According to their misattribution, the subsequent reaction of the organizational participants may further exacerbate the situation of the vicious cycle. This research suggests that the technology learning trap and implementation effectiveness trap result from not only the characteristics of technology (in this case, net-centric systems) and the inherent structure of the implementation process (i.e., the balancing and reinforcing feedback loops), but the dynamic interactions between human agency and the technology implementation process (Orlikowski 1992).

Managers who make the decision of net-centric implementation for their organizations need to be aware and prepared of the potential traps in the implementation process. Long-term commitment is definitely helpful to net-centric implementation. Patient and consistent decision-makings about the tradeoff between short-term performance drop and potential long-term benefits and the tradeoff between local project needs and organization-level net-centric implementation are important. In addition, institutionalization of the long-term commitment and consistent decision-making about the tradeoffs using appropriate governance mechanisms may increase the chance of successful net-centric implementation (Joachim et al. 2011; Varadan et al. 2008). The traps discussed here can be of even greater significance in defense and intelligence systems where unique operational and security considerations require extensive, often multi-year, customization of commercial SOA software before benefits can begin to be seen.

References

Armstrong, C.P., and Sambamurthy, V. 1999. "Information Technology Assimilation in Firms: The Influence of Senior Leadership and It Infrastructures," *Information systems research* (10:4), pp 304-327.

Association for Enterprise Integration (AFEI). 2008. Industry Recommendations for DoD Acquisition of Information Services and SOA Systems. SOA Acquisition Working Group, AFEI Executive Forum on Business Change, Arlington, Va., July 7."

Arrow, K.J. 1962. "The Economic Implications of Learning by Doing," *The Review of Economic Studies* (29:3), pp 155-173.

Baskerville, R., Cavallari, M., Hjort-Madsen, K., Pries-Heje, J., Sorrentino, M., and Virili, F. 2005. "Extensible Architectures: The Strategic Value of Service Oriented Architecture in Banking," *ECIS 2005 Proceedings*), p 61.

Bieberstein, N., Bose, S., Walker, L., and Lynch, A. 2005. "Impact of Service-Oriented Architecture on Enterprise Systems, Organizational Structures, and Individuals," *IBM Systems Journal* (44:4), pp 691-708.

Boh, W.F., and Yellin, D.M. 2010. "Enablers and Benefits of Implementing Service-Oriented Architecture: An Empirical Investigation," *International Journal of Information Technology and Management* (9:1), pp 3-29.

Borges, B., Holley, K., and Arsanjani, A. 2004. "Service-Oriented Architecture: Components and Modeling Can Make the Difference," *Web Services Journal* (9:1), pp 34-38.

Cherbakov, L., Galambos, G., Harishankar, R., Kalyana, S., and Rackham, G. 2005. "Impact of Service Orientation at the Business Level," *IBM Systems Journal* (44:4), pp 653-668.

Choi, J., Nazareth, D.L., and Jain, H.K. 2010. "Implementing Service-Oriented Architecture in Organizations," *Journal of Management Information Systems* (26:4), pp 253-286.

Croom, C. 2006. "Service-Oriented Architectures in Net-Centric Operations," *CROSSTALK The Journal of Defense Software Engineering*, July 2006, pp 13-15.

DiMaggio, P.J., and Powell, W.W. 1983. "The Iron Cage Revisited: Institutional Isomorphism and Collective Rationality in Organizational Fields," *American sociological review* (48:2), pp 147-160.

Fricko, A. 2006. "Soas Require Culture Change and Service Reuse," *Business Communications Review* (36:5), p 58.

Georgantzas, N.C., and Katsamakas, E.G. 2008. "Information Systems Research with System Dynamics," *System Dynamics Review* (24:3), pp 247-264.

Grant, D., Hall, R., Wailes, N., and Wright, C. 2006. "The False Promise of Technological Determinism: The Case of Enterprise Resource Planning Systems," *New Technology, Work and Employment* (21:1), pp 2-15.

Hau, T., Ebert, N., Hochstein, A., and Brenner, W. 2008. "Where to Start with Soa: Criteria for Selecting Soa Projects," *Proceedings of the 41st Hawaii International Conference on System Sciences (HICSS 2008)*, Waikoloa, Big Island, Hawaii: IEEE Computer Society, pp. 314-322.

Joachim, N., Beimborn, D., and Weitzel, T. 2011. "What Are Important Governance and Management Mechanisms to Achieve It Flexibility in Service-Oriented Architectures (Soa)?: An Empirical Exploration," *Proceedings of the 44th Hawaii International Conference on System Sciences (HICSS 2011)*, Kauai, Hawaii USA: IEEE Computer Society, pp. 1-10.

Kanaracus, C. 2011. "Soa Is Alive and Well, Forrester Says." Computer World.

Kotter, J. 1995. "Leading Change: Why Transformation Efforts Fail," *Harvard Business Review* (73), pp 59-67.

Krafzig, D., Banke, K., and Slama, D. 2005. *Enterprise Soa: Service-Oriented Architecture Best Practices*. Prentice Hall PTR.

Law, J., and Urry, J. 2004. "Enacting the Social," *Economy and Society* (33:3), pp 390-410.

Lee, J.H., Shim, H.J., and Kim, K.K. 2010. "Critical Success Factors in Soa Implementation: An Exploratory Study," *Information Systems Management* (27:2), pp 123-145.

Liang, H., Saraf, N., Hu, Q., and Xue, Y. 2007. "Assimilation of Enterprise Systems: The Effect of Institutional Pressures and the Mediating Role of Top Management," *Management Information Systems Quarterly* (31:1), p 6.

Luthria, H., and Rabhi, F. 2008. "Organizational Constraints to Realizing Business Value from Service Oriented Architectures: An Empirical Study of Financial Service Institutions," *Service-Oriented Computing - ICSOC 2008*), pp 256-270.

Luthria, H., and Rabhi, F. 2009a. "Service Oriented Computing in Practice: An Agenda for Research into the Factors Influencing the Organizational Adoption of Service Oriented Architectures," *Journal of Theoretical and Applied Electronic Commerce Research* (4), pp 39-56.

Luthria, H., and Rabhi, F. 2009b. "Using Service Oriented Computing for Competitive Advantage," *Proceedings of Americas Conference on Information Systems (AMCIS 2009)*, p. 140.

McKendrick, J. 2011. "So Far, Soa Failures Are Few and Far between, Survey Says." ZDNet.

Moitra, D., and Ganesh, J. 2005. "Web Services and Flexible Business Processes: Towards the Adaptive Enterprise," *Information & Management* (42:7), pp 921-933.

Mueller, B., Viering, G., Legner, C., and Riempp, G. 2010. "Understanding the Economic Potential of Service-Oriented Architecture," *Journal of Management Information Systems* (26:4), pp 145-180.

Orlikowski, W.J. 1992. "The Duality of Technology: Rethinking the Concept of Technology in Organizations," *Organization Science* (3:3), pp 398-427.

Pfeffer, J. 1997. "New Directions for Organization Theory:: Problems and Prospects,").

Pfeffer, J., and Sutton, R.I. 2000. *The Knowing-Doing Gap: How Smart Companies Turn Knowledge into Action*. Cambridge, MA, USA: Harvard Business School Press.

Repenning, N.P. 2002. "A Simulation-Based Approach to Understanding the Dynamics of Innovation Implementation," *Organization Science* (13:2), pp 109-127.

Repenning, N.P., and Sterman, J.D. 2001. "Nobody Ever Gets Credit for Fixing Problems That Never Happened," *California management review* (43:4), pp 64-88.

Repenning, N.P., and Sterman, J.D. 2002. "Capability Traps and Self-Confirming Attribution Errors in the Dynamics of Process Improvement," *Administrative Science Quarterly* (47:2), pp 265-295.

Rosenberg, N. 1982. *Inside the Black Box: Technology and Economics*. Cambridge: Cambridge University Press.

Rudolph, J.W., Morrison, J.B., and Carroll, J.S. 2009. "The Dynamics of Action-Oriented Problem Solving: Linking Interpretation and Choice," *The Academy of Management Review (AMR)* (34:4), pp 733-756.

Ryan, G.W., and Bernard, H.R. 2000. "Data Management and Analysis Methods," in: *Handbook of Qualitative Research*, Y.S.L. Norman K. Denzin (ed.). Sage, CA, USA: Citeseer, pp. 769-801.

Scott, S.V., and Wagner, E.L. 2003. "Networks, Negotiations, and New Times: The Implementation of Enterprise Resource Planning into an Academic Administration," *Information and organization* (13:4), pp 285-313.

Sher, P.J., and Lee, V.C. 2004. "Information Technology as a Facilitator for Enhancing Dynamic Capabilities through Knowledge Management," *Information & Management* (41:8), pp 933-945.

Śliwaa, J. and Amanowicz, M. 2011. "Success Factors for SOA Implementation in Network Centric Environment," *Journal of Telecommunications and Information Technology* 1/2011, pp 43-53. http://www.nit.eu/czasopisma/JTIT/2011/1/43.pdf

Smith, R. 2008. "A Simpler Approach to Soa." Information Week

Sterman, J.D. 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World with Cd-Rom*. Irwin/McGraw-Hill.

Tversky, A., and Kahneman, D. 1974. "Judgment under Uncertainty: Heuristics and Biases," *Science* (185:4157), p 1124.

Varadan, R., Channabasavaiah, K., Simpson, S., Holley, K., and Allam, A. 2008. "Increasing Business Flexibility and Soa Adoption through Effective Soa Governance," *IBM Systems Journal* (47:3), pp 473-488.

Volkoff, O., Strong, D.M., and Elmes, M.B. 2007. "Technological Embeddedness and Organizational Change," *Organization Science* (18:5), pp 832-848.

Vroom, V.H. 1964. *Work and Motivation*. New York: Wiley.

Wheelwright, S.C., and Clark, K.B. 1995. *Leading Product Development*. Free Press.

Wood, R., and Bandura, A. 1989. "Social Cognitive Theory of Organizational Management," *The Academy of Management Review* (14:3), pp 361-384.

Yoon, T., and Carter, P. 2007. "Investigating the Antecedents and Benefits of Soa Implementation: A Multi-Case Study Approach," *AMCIS 2007 Proceedings*), pp 195-205.

PART III

Effect of Funding Fluctuations on Government Funded Software Development

Travis Trammell, Stuart Madnick, Allen Moulton
MIT Sloan School of Management
77 Massachusetts Avenue
Cambridge, MA 02442

Introduction

The size and scope of the United States defense budget is enormous. A great deal of political attention is paid to such a large amount of government spending, with a particular focus on maximizing the value obtained for taxpayer dollars and minimizing waste. The Department of Defense is in a constant state of self assessment in search of areas to reduce cost and improve efficiency. However, very little research has been conducted to investigate the effects of fluctuations in government budgets on the effectiveness of software acquisition. Although it is well understood that budget fluctuations have an impact, it has been difficult to understand the magnitude of the impact and all the causes. The impact of these fluctuations on government sponsored software development will be examined with a focus on changes to productivity and product delivery delay.

This paper will begin by discussing some of the reasons for concern with the delays in production that often accompany these budget fluctuations. These include: first mover advantage, macroeconomic risk, and the inability to meet requirements. Having established some of the negative implications resulting from product delay, a brief description of prior work in this field will follow. Next, a significant amount of time is dedicated to describing the system dynamics model and the results generated from that model. The insights gathered from these results will provide decision makers with further considerations for the negative impacts of budget fluctuations. Two key insights include the existence of a "ramp up tax" within projects that slows development and the loss of skill and familiarity with projects when employees are not working on those projects. This discussion analyzes the many elements within the process that can produce a change in budgets from year to year and notes the different impact of temporarily stopping a project versus reducing the funding level on the project. Finally, a conclusion will summarize the findings and recommend areas of follow on research.

The United States Government Budgeting Process and Factors Contributing to Budget Uncertainty

Multiple Players in the Budget Process

There are many players and stages in the Department of Defense Budget process, these include (1) the Presidential Budget Request[1] , (2) the Congressional Budget Process[2], (3) Continuing Resolutions[3], and (4) Supplemental Appropriations.[4] As a result of the uncertainty created at each of these stages or events, long-term funding is difficult, especially for large-scale complex software development efforts that often extend over multiple years. This section is not intended to be comprehensive nor complete but to give a general idea of the complexities and uncertainties.

---

[1] Executive Office of the President, Office of Management and Budget, Circular Number A-11, June 2008

[2] The Congressional Budget Process-A Brief Overview by James V. Saturno, Specialist on the Congress Government and Finance Division

[3] http://thisnation.com/question/003.html

[4] The Cost and Funding of the Global War on Terror (GWOT), Testimony before U.S. House of Representatives, Committee on the Budget, Steven M. Kosiak, Director of Budget Studies

In addition, to the on-going causes mentioned in the previous paragraph, there are some additional factors that further complicate the situation.

Historical Decline Following Major Conflicts

Operation Enduring Freedom, the U.S. War against the Taliban in Afghanistan, is approaching the 10 year mark and is already the longest war in American history. The cost of the Global War on Terrorism has totaled approximately $1.28 trillion as of March 2011.[5] Given the tremendous expenditure over the last 10 years contraction of military spending appears inevitable. Historically, there has been an approximately 30% reduction in defense related spending following major conflicts that involved the United States.[6]

Deficit Reduction

With the U.S. debt approaching near historic levels, budget cuts have occupied a great amount of the legislative conversation. In an attempt to preempt larger defense cuts, the DOD has committed to cutting approximately $464 billion over the next ten years.[7] Although the details of these cuts are yet to be released, many DOD projects will likely be affected and endure reduced funding levels. This represents an approximately seven percent cut per year from the DOD budget topline. Additionally, as part of the debt ceiling increase deal of 2011, if the Congressional Super Committee (or subsequent Congress actions) fails to agree to cuts totaling $1.5 trillion, the DOD will receive an additional $750 billion cut over the next 10 years. Should this situation occur, most DOD programs would receive some level of cuts.

Other Government Agency Level Budget Causes of "Gaps" or "Reductions"

Government Contracting Regulations -> "Gaps"

Government contracts are awarded for periods of performance, most commonly one year. Funds allocated for a certain period of performance are required to be spent during that period of performance. This can create gaps in funding between the different periods of performance. During these gaps, work cannot be conducted on the specific project.

Furthermore, government contracts are limited to five years by government regulation. Thus, a new bidding process must be conducted if work beyond five years is required on a specific project. This rebidding process can be complicated and contentious, which can also result in a funding gap.

Interagency "Tax" Budget Transfers -> "Reductions"

An agency can lose some of its yearly funding in the form of a "tax" from higher levels in the government hierarchy. This gives the higher level influence into the conduct and focus of the lower level agency's operations but produces budget uncertainty for its projects. The scale of this "tax" is typically around 15% so the agency might have to operate at a lower funding rate. Often, a significant portion of this"tax" is returned to the agency later in the fiscal year for spending within that year. Unfortunately, this is not guaranteed nor easy to predict, so there is often need for adjustments during the year depending upon whether the funding is increased or not.

---

[5] The Cost of Iraq, Afghanistan, and Other Global War on Terror Operations Since 9/11, Amy Belasco, Specialist in U.S. Defense Policy and Budget, March 29, 2011

[6] Defense Strategy in a Time of Budget Austerity, Kathleen Hicks, Deputy Under Secretary of Defense for Strategy, Plans and Forces, U.S Department of Defense presented on 14 September 2011

[7] Ibid.

Prior Research on Impact of Product Delivery Delay

Project gaps or funding reductions definitely will affect the completion date of a software development project. But, determining the magnitude of the impact can be difficult as noted in reports such as by Ronald Kadish et al (2006) This has important consequences in both the private and public sector for multiple reasons.

Time to Market

The length of time it takes for a software product to reach the market has several implications for the developer. The first is that the overall cost of the program is increased. As work stretches out, labor and development costs continue. Additionally, the old system must be maintained for a longer period of time. This maintenance is often more expensive because it occurs as the end of the useful life of the out-of-date system. Finally, while the new system is under development, the organization is not receiving the expected benefits from the system. This induces a productivity loss on the organization because it is forced to operate with less capable systems.

First Mover Advantage

The implications for delaying a product launch are well documented and have been studied at length. A common reference to such an incentive is the "first mover advantage". The first mover advantage refers to the considerable incentives that exist for being first to market with a new product. Lieberman and Montgomery (1988) describe three primary drivers of first mover advantage: technological leadership, preemption of assets, and buyer switching costs. Of these three sources of advantage for the first mover, when government work is examined, buyer switching costs appear to carry the most weight. It is common parlance in government circles that the most difficult thing to accomplish in acquisition is to end an established program. The well documented struggles by the Department of Defense to end the Joint Strike Fighter second engine program provides for an illustrative example. From 2006 to 2011, the Department of Defense vigorously appealed for the discontinuation of the second engine produced by General Electric. The causes of this so called "refusal to die" by projects are not the focus of this research. This discussion just accepts this staying power as given and considers the resulting implications for buyer switching costs. In the normal interpretation of buyer switching costs, the direct costs of moving to a new product or system are examined. For example, the cost of purchasing the new system, retraining the workforce, productivity loss during the transition, and failure risk. In the government arena, all of these factors matter greatly but are also increased because of the organizational inertia previously discussed. The first mover advantage takes on even greater importance in the government sector as a result.

An interesting case study for the effect of this governmental first mover advantage is the Defense Intelligence Agency's (DIA) Human Intelligence (HUMINT) Online Tasking and Reporting (HOT-R) compared to the Combined Information Data Network Exchange (CIDNE). CIDNE became a very specialized tool developed out of the frustrations of commanders in Iraq about the inability to share operational information or gain a more complete understanding of the situation on the ground. By government comparison, CIDNE was created and launched in record time to generally positive responses. Most importantly CIDNE garnered significant support from senior decision makers in the U.S. Central Command, most notably General David Petraeus. While CIDNE performed well as an operational tool and was valuable in maintaining intelligence archives in theater, it was not designed as a HUMINT reporting system. The Defense Intelligence Agency (DIA), acting as the lead executive agent within the U.S. government for HUMINT, was tasked with standardizing HUMINT reporting across the intelligence community. In pursuit of this objective, DIA developed HOT-R. This system focused solely on HUMINT reporting and did not replicate any of the CIDNE functionality in the operational realm. However, HOT-R has experienced significant resistance from within U.S. Central Command because of the perception that it duplicates or is meant to replace CIDNE in some way. Outside of Central Command, the HUMINT community has seen HOT-R as superior product upgrade to procedures previously in use. The key issue is the order of system development. It appears plausible that had HOT-R been developed first, all the community of users would have accepted the system. However, CIDNE had the first mover advantage in this case. This speaks to the effect of switching costs in the government and the impact they play on acquisition decisions.

Macroeconomic Risk

A second major incentive for the government to get products to market faster and in the hands of customers is the issue of risk. From the standpoint of macroeconomic and business cycle effects, it is advantageous to complete projects as expeditiously as possible. Stretching a project out over longer and longer periods will make budgeting for the project much more difficult and expose the project to more economic risk. Government revenue is directly affected by the business cycle and government projects are generally less stable under periods of economic hardship.

Inability to Meet Requirements

Finally, the longer a project takes to reach the market, the greater the likelihood that the original requirement for the product has changed. The pace of technological advancement continues to increase along with the introduction of disruptive technologies. This risk continues to increase as newer and more innovative products are developed. A similar pace of change has been seen within the security community. During the Cold War, the focus of security and intelligence community did not change greatly and thus systems could have a much longer development time. Given the dynamic security environment that exists now and likely in the future, products that take too long to develop will not arrive in time to meet the threat that was the impetus for the original development.

System Dynamics Modeling of Software Development

System Dynamics

The System Dynamics (SD) approach, as explained in John Sterman (2000), is based on identifying individual causalities and how they combine to create, often non-linear, feedback loops that can be the causes of counter-intuitive outcomes. This approach can greatly leverage the deep, but often isolated and fragmented, knowledge. The core of the SD modeling strategy is representation of system structure in terms of stocks, flows, and the causal mechanisms that govern their rates of change. In this connection, feedback loops are the building blocks for articulating the causality represented in these models. The interaction among the various feedback loops in a model can represent and explain system behavior. SD models offers unique capabilities to the understanding since it explicitly recognizes the complex interactions among many feedback loops, rejects notions of linear unidirectional cause-and-effect, and allows the analyst to view a complete system of relationships whereby the 'cause' might also be affected by the 'effect' and enables analysts to uncover 'hidden' dynamics. It also allows for understanding the long term effects of short term policies or behaviors.

Additional unique capabilities of SD include: (1) *Objective input*: Ability to utilize data to determine, with precision, parameters affecting the causality of individual cause-and-effect relationships, (2) *Subjective (expert) judgment*: Ability to represent and model cause-and-effect relationships, based on expert judgment, even when detailed data does not exist, (3) *Intentions Analysis*: Ability to identify the long-term unintended consequences of policy choices or actions taken in the short term, and (4) *Tipping point analysis*: Ability to identify and analyze "tipping points" – where further incremental changes lead to significant impacts, and (5) *Transparency*: Ability to explain the reasoning behind predictions and outputs of the SD model.

System Dynamics has been used to model a diverse array of situations, including crisis in the world oil market, dynamics of arms races, threats to sustainability, combat vehicle accidents (in non-combat settings), impact and usage of social media on uprisings, and stability and instability of countries. An example of the latter can be found in Choucri, N. Electris, D., Goldsmith, D., Mistree, D., Madnick, S., Morrison, J.B., Siegel, M., Sweitzer-Hamilton, M. (2006),

Prior System Dynamics Model of Software Development Performance

Although this project offers insights for the greater DOD and government community, the focus of this project has been software development. As a result, the model relies greatly on prior research conducted on the factors effecting software development productivity. This research provides fidelity and explanation for these factors. The model described here is based on the Software Development

Performance Model (SDPM) documented in Abdel-Hamid and Madnick (1991). Although certain management actions that led to variations in total project cost were studied in SDPM, changes in budget constraints was not a central focus of that work.

Early on in the examination of this issue, the intuition was that the effect of budget fluctuations would be most directly seen in staffing adjustments. As a result, the portion of the SDPM model, shown in Figure 1, that described staffing was extracted and used as a guide for developing the new model. The initial values for many of the variable were also based on those used in the SDPM model and confirmed through the interview process.

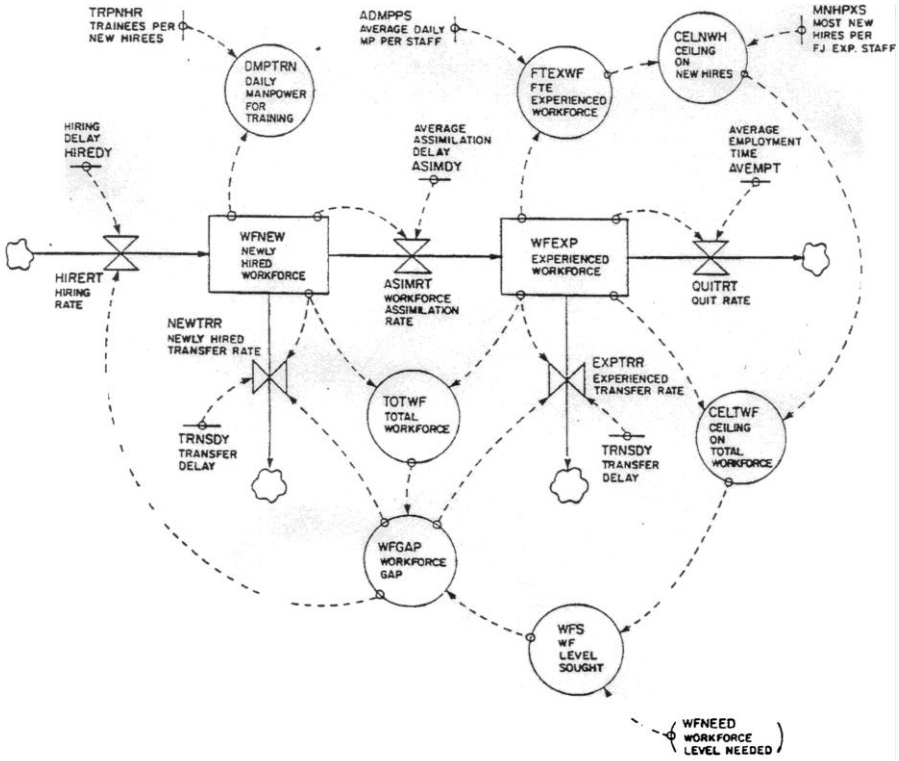


Figure 1. Staffing Model Subsection from Abdel-Hamid and Madnick (1991), p.64.

Government Funded Software Development Performance Model

Model Overview

The original SDPM model had four major subsections: (1) Project Staff Provisioning, (2) Project Staff Planning, (3) Funding Process and Expenditure Accounting, and (4) Software Development. The design of our Government Funded Software Development Performance Model (GFSDPM) focused on the Project Staff Provisioning aspect. As a consequence, the remaining subsections of the model were simplified in order to isolate the effects of these budget fluctuations. Figure 2 shows the overall organization of GFSDPM model and its four subsections.

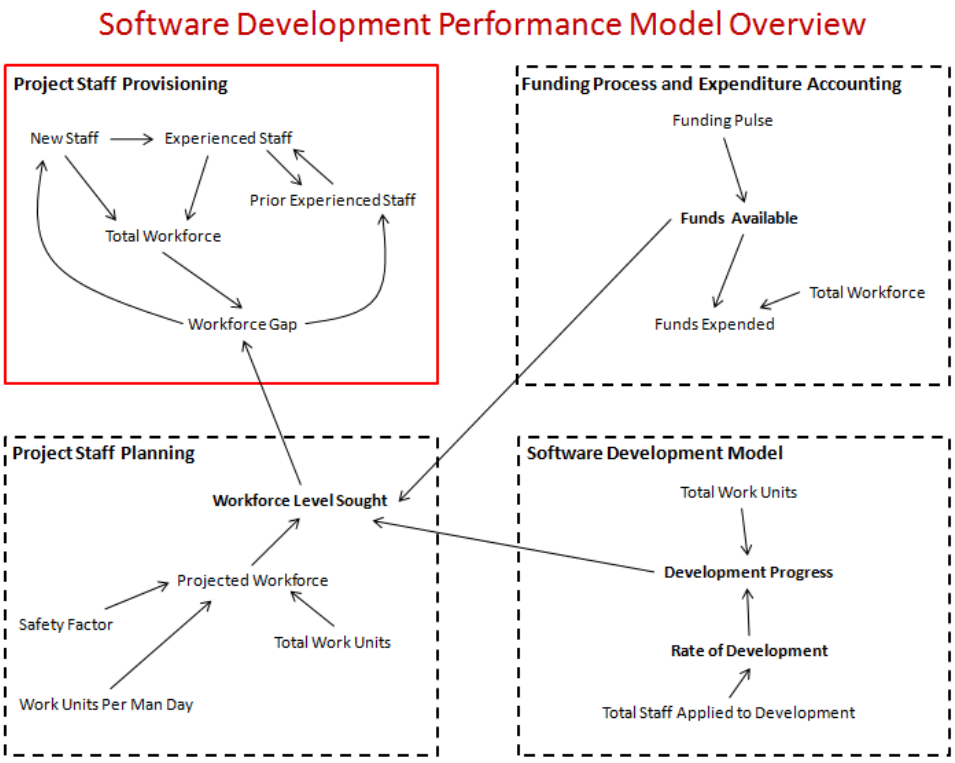


Figure 2. The Four Model Subsections

The model illustrated above performs in the following manner. The Software Development Model begins with a required amount of work to be accomplished. It then compares that with the amount of work already accomplished to determine the remained amount of work to be completed. In the Project Staff Planning section, based on the due date of the project, the model determines the required amount of staff to complete the project on time. It then checks if funding is available for the "Workforce Level Sought." The Funding Process and Expenditure Accounting determines how much funding is available. If sufficient funding is not available, the "Workforce Level Sought" amount that is sent to Project Staff Provisioning section is reduced. From there, the Project Staff Provisioning section conducts the hiring and training which will be discussed next. This is a dynamic model, so these steps are repeated at each time interval, as illustrated later.

Enhanced Project Staff Provisioning Model Subsection

Our enhanced project staff provisioning model subsection is shown in Figure 3 and explained below.

The key addition to the original SDPM model incorporated into the Project Staff Provisioning subsection is the "Prior Experienced Staff" stock. The singular factor that emerged from interviews with agency staff in regards to staffing was the length of time required to get new personnel cleared to work on a project. This clearance process consisted of two major steps. The first was the standard U.S. government background investigation. This process consists of an extensive background investigation and is mandatory for all personnel with access to Secret or Top Secret material which can take up to a year and a half depending on factors. Furthermore, for this agency[8], all software development is conducted within secure areas that require a clearance to access. Additionally, contractors[9] have a lower priority for investigations than do military personnel or government employees. The investigation process is also

[8] The parameters and situation used were based on detailed interviews with a particular government agency. Although these factors might vary in other agencies, the general analysis is broadly applicable.

[9] In most cases, development work on these projects is done by contractor personnel.

affected by funding considerations. For example, in May of 2011, all investigations were placed on hold due to funding uncertainty during Congressional Budget negotiations. Once an employee finally receives a clearance, he or she must still receive building access, referred to as getting "badged," to the agency's facilities. This adds additional delay to acquiring new personnel.

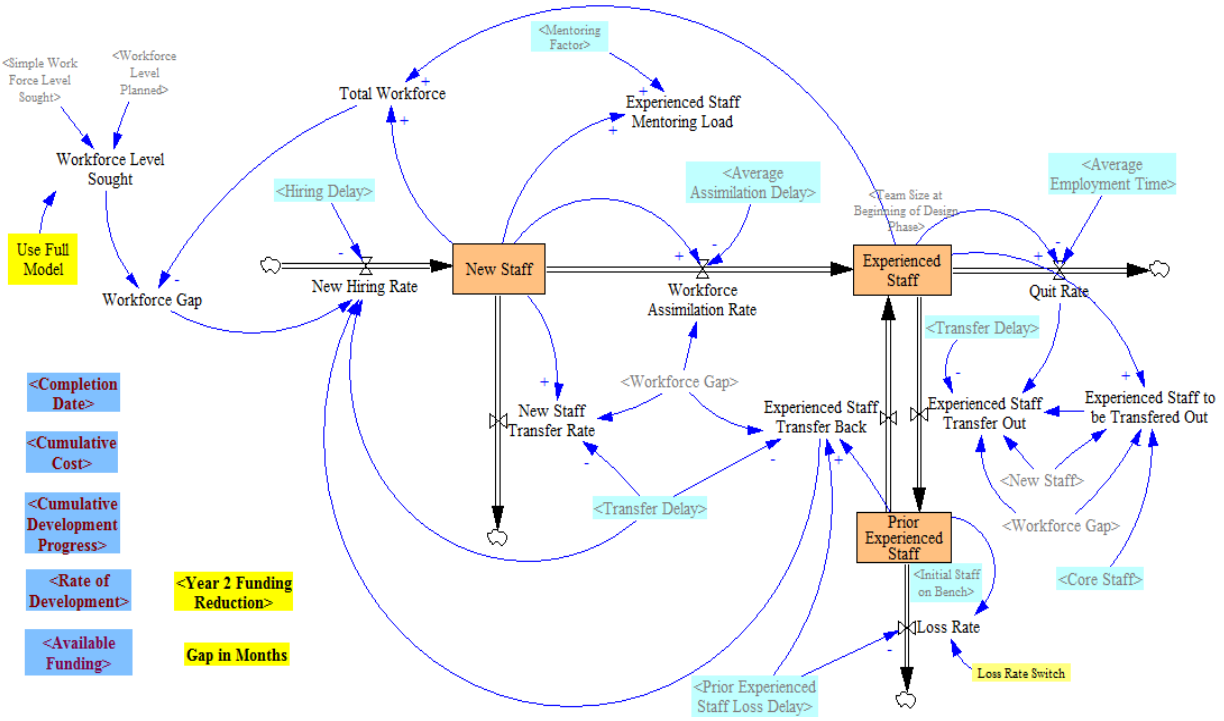


Figure 3. Enhanced Project Staff Provisioning Model Subsection

The net result of this process is that personnel who have completed this process and received a "badge" become very valuable. Great efforts are taken to avoid the loss of these personnel. One of the most common methods to avoid the loss of these personnel is to rotate them among different projects when funding for one project if reduced or exhausted. This method is referred to as placing employees "on the bench". If employees were not placed on the bench, they would lose their badge for building access and bringing them back on to a previous project would be greatly delayed. The group of employees placed on the bench during periods of reduced funding is the "Prior Experienced Staff" stock shown in Figure 3.

While on the bench, these employees do experience a loss of familiarity with the specific project and possibly a loss of overall skills, as explained in more detail below. For purposes of our experiments with the model, the "Loss Rate" of project specific knowledge for the "Prior Experienced Staff" is assumed to occur continuously over 80 days. This value was suggested by Subject Matter Experts (SMEs) as being the same length of time as they estimated that it takes "New Staff" to assimilate and become "Experienced Staff," shown as the Average Assimilation Delay in Figure 3. Another important feature of the Staff Provisioning Model is that when funding is reduced, the first employees to be let go are the new employees. Similarly, when funding is increased, employees are drawn from the "Prior Experienced Staff" first. The number of people on the bench is also reduced by a small number of employees quitting the organization while on the bench, shown as the Prior Experienced Staff Loss Rate in Figure 3.

Funding Process and Expenditure Accounting

One factor that was absent from the SDPM model is the possibility of funding fluctuations. The SDPM model was primarily geared toward commercial software development, without consideration of budget fluctuations. The basic idea was that as long as a commercial project is meeting expectations, it will remain funded unless the project is modified for some other reason. In our enhanced model, the

Funding Process and Expenditure Accounting portion of the model are designed to produce some of the funding scenarios seen by the developer team. These include: (1) continuous funding, (2) funding gaps of varying lengths, and (3) reduction in funding that does not produce a funding gap.

The Project Staff Planning portion of the model is also very simple in design. This section of the model produces the staffing level necessary to complete the project based a number of work units required for the project and the desired completion date of the project. The model attempts to maintain this staffing level as long as funding is available.

The Software Development model, as based on the original SDPM model, takes into account the fact that new staff are on average about half as effective as experienced staff[10]. It also addresses the reality that the amount of work conducted by experienced staff is reduced by the need to train new staff.

Experiment Design

As previously mentioned, funding levels directly impact the amount of staff assigned to the project. The base case for our experiments was set to a staff number of 20 workers. The level of work required was set to be completed in exactly two years, that is, 480 working days in the model. For this experiment, we assumed that the funding adjustments occurred during Year 2 only. In Year 3, the Year 1 staffing level of 20 workers is resumed. All of the funding test cases are compared based on the amount of spillover of work into the third year that results from funding modifications. The overall experimental design is shown in Figure 4.

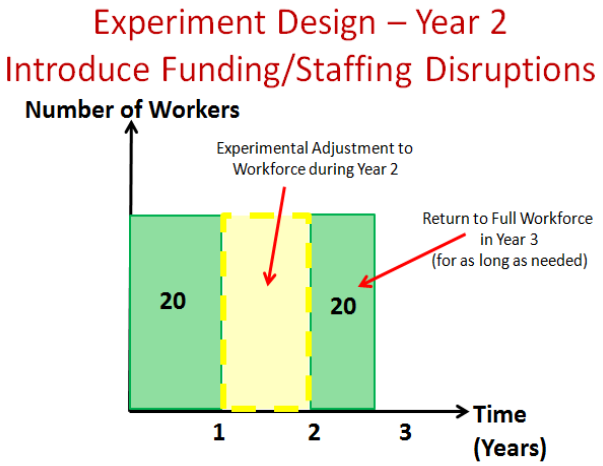


Figure 4. Experiment Design

Test Cases and Results

No Funding Gaps or Funding Reductions

The No Gap case, shown to the left in Figure 5, is the baseline test in which sufficient funding is provided to complete the project within two years. This funding amount corresponds with 20 employees. The days referenced are work days. There is assumed to be five works days in a week, four weeks in a month, and twelve months in a year. Thus, the end of year one occurs at day 240, year two at day 480, and year three at day 720.

[10] "Half" was a typical estimate provided by the Subject Matter Experts. The SD model is focused on understanding overall behavior, the precise value of these parameters does not change the overall behavior. The model can be run using different parameter values to reflect the precise details of a specific organization.

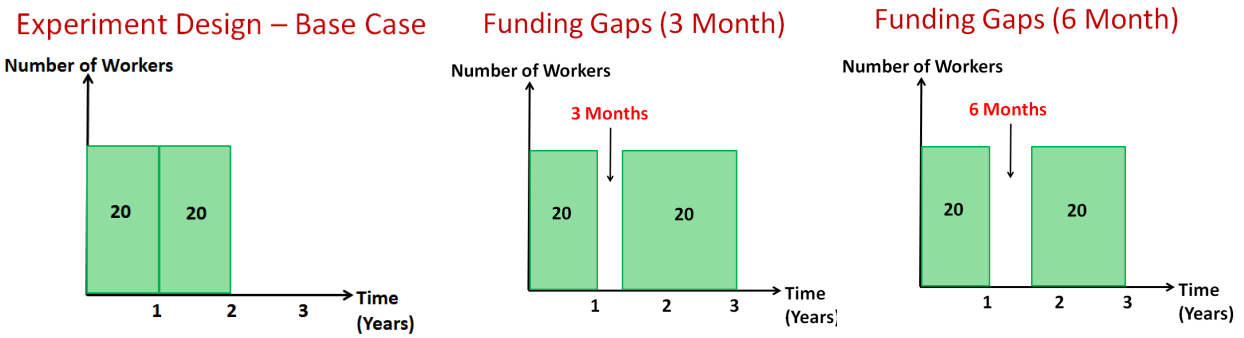


Figure 5. Three Gap cases: No Gap, 3 month Gap, 6 month Gap

Three Month and Six Month Funding Gap

These two additional test cases consider a three month or six month funding gap between the first and second year of the project, as shown in the center and right side of Figure 5. During this gap, all experienced staff are transferred to the bench and any new staff are released. This results in no work being conducted on the project during the gap.

Reduction of Funding by 25% and 50% in the Second Year

We also considered any type of funding fluctuation, that is a reduction in funding. These cases model a 25% or 50% reduction of funding in the second year of the contract. For example, if the first year funding was $5 million, for the 25% reduction case, the second year funding is reduced to $3.75 Million. The reduction of 25% was chosen in order to compare the results to a three month funding gap. Assuming constant funding over the entire year, a three month funding gap reduces spending by 25% for that year. The funding reduction results in a commiserate reduction in workforce (15) during the second year as illustrated in the center case of Figure 6. The third case, shown on the right side of Figure 6, models a 50% reduction of funding in the second year of the contract, that is, second year funding is $2.5 million and this reduction will be compared with the six month gap. The reduction reduces the workforce to 10 in the second year.

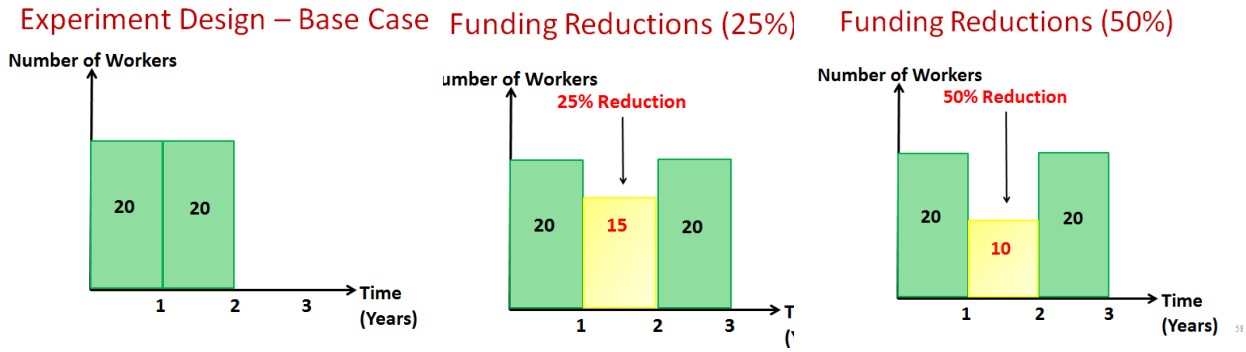


Figure 6. Three Reduction cases: No Reduction, 25% Reduction, 50% Reduction

Test Results

There were several interesting insights gained by developing and running our model for these various cases. These insights include the discovery and analysis of what we refer to as the "Ramp Up" tax and the "Gap Tax," as explained below

Ramp Up Tax

In our conversations with developers and development managers who provided much of the structure for our model, they noted that, the rate of development took time to ramp up and then stabilize. This effect is referred to as the "ramp up tax" in all subsequent discussion. The ramp up tax is a result of the time taken to hire new staff[11] combined with the time taken for these new staff members to assimilate on to the project. The Subject Matter Experts that we interviewed suggested a shape that lead to Figure 7[12].

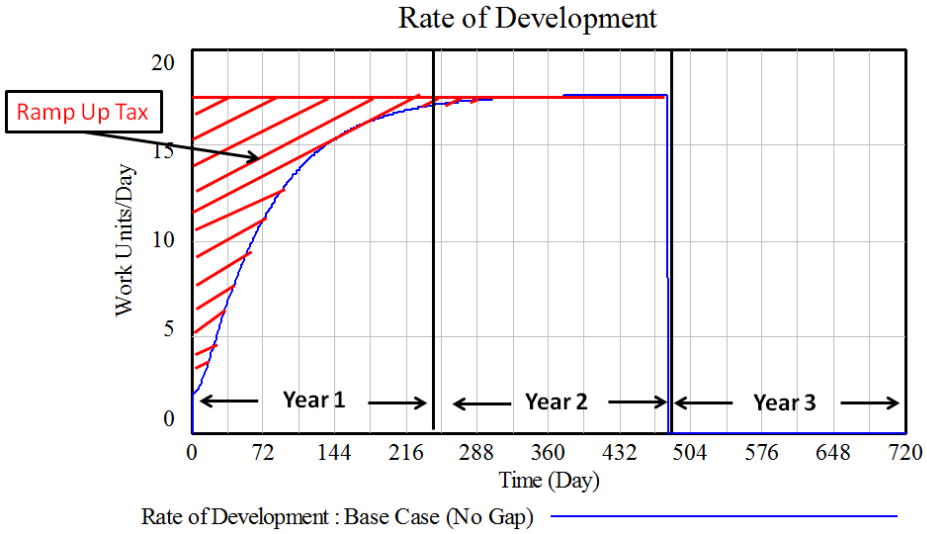


Figure 7. The "Ramp Up" tax

Assimilation

Assimilation is an important concept, it refers to the time it takes for an employee newly assigned to a project to become familiar with that project. We assume that the new hire is a trained professional but is new to this particular project – thus some time must be spent to learn the goals of the project, the current status of the project, project specific knowledge (e.g., software standards and libraries to be used), his/her specific responsibilities, etc. The ramp up tax is further influenced by the fact that due to this lack of project specific knowledge, new staff are half as effective as experienced staff. Furthermore, there is a need for experienced staff to spend significant time helping to assimilate new staff, which reduces the productivity of the Experienced Staff. The chart shown in Figure 7 illustrates the ramp up tax in the no funding gap base case. This chart shows the rate of development in work units/day plotted over the life of the project.

Due to the delays in hiring staff and the assimilation process, it takes about a year for the team to attain its stable level of productivity, as shown in Figure 7. The upper left corner area, shaded in red, is what we refer to as the "Ramp Up Tax." In the ideal case, with a team of 20 people, we might expect the steady state productivity level to be 20 work units/day, but even in steady state, there are people that leave the project and new hires being added – which continuously reduces the productivity somewhat to a level of about 17 work unuts/day.

---

[11] Note: We only consider the time after a person has received his or her "badge," whether that is a new person for the agency or someone transferred from another project within the agency. Due to the lengthy security clearance process, most people, after receiving their badge, are moved around within the agency to avoid needing to go through the extensive delay of the security clearance process. There is some modest delay caused by the procedures involved in changing assignment from one project to another project, so it is not really "hire new staff," in most cases it is "reassign new staff."

[12] Of course, different projects have slightly different shapes. Figure 7 was intended to be a representative example for purposes of our experiments. It could be modified to reflect the details of a specific project.

Skills Loss Effect

While on the bench, employees experience skill loss and loss of familiarity with the project. Employees experience this loss of familiarity with the project because while on the bench they are actually assigned and working on other projects. As a result, the focus and daily involvement that is required for employee effectiveness on complex projects does not occur while employees are on the bench. The longer the employees remain on the bench the greater the loss of skills and familiarity with the project. This results in a reduced rate of development after a funding gap, as shown in Figure 8.

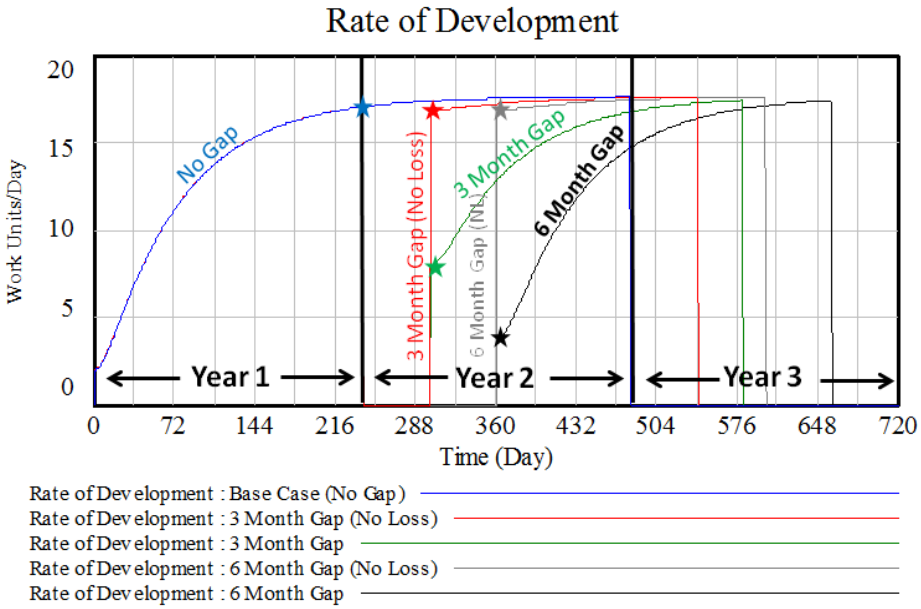


Figure 8. Affect of Funding Gap on Rate of Development

Ignoring Skills Loss: The Rate of Development as indicated in Figure 8 drops only slightly when no skills loss is modeled. In that case, the drop in the Rate of Development following a gap only occurs because the quit rate still reduces the "Experienced Staff" stock, though only slightly.

Including Skills Loss: When the skills loss effect is modeled, funding gaps significantly reduce the productivity of the workforce when work is resumed after the gap. Additionally, as the size of the funding gap increases, the reduction in the Rate of Development following the gap increases as well. In other words, the longer an employee sits on the bench, the more skills loss they suffer. The longer the funding gap, the larger the productivity loss following the gap.

The decreases in productivity are summarized in Table 1. In the base case, where there is no funding gap, the team starts the second year with its steady state productivity of 17.06 work units per day, but this starting productivity for the second year drops to 8.08 or 3.76 work units per day after a 3 month or 6 month gap, respectively.

Table 1. Affect of Funding Gap on Rate of Development

Rate of Development Comparison (Work Units Per Day)		
Test Case	Productivity Level at start of work after Gap in Year 2	% Lost Productivity vs No Gap Full Funding
Base Case (No Gap)	17.06	0%
3 Month Gap	8.08	53%
6 Month Gap	3.76	78%

Affect of Funding Gap on Development Progress

The reduction in development productivity caused by those funding gaps delays completion of the project, as shown in Figure 9, beyond just the length of the actual funding gap time. This is in contrast to the naïve, though frequent assumption, that a 3 month funding gap merely delays the completion of a project by 3 months. We refer to this additional delay as the "Gap Tax." The longest delay, for these cases, is seen from the 6 Month Gap which also saw the largest drop in Rate of Development following the gap. The size of these additional delays in completion, beyond the funding gap itself, that is the "Gap Tax," are listed in the right-most column of Table 2.

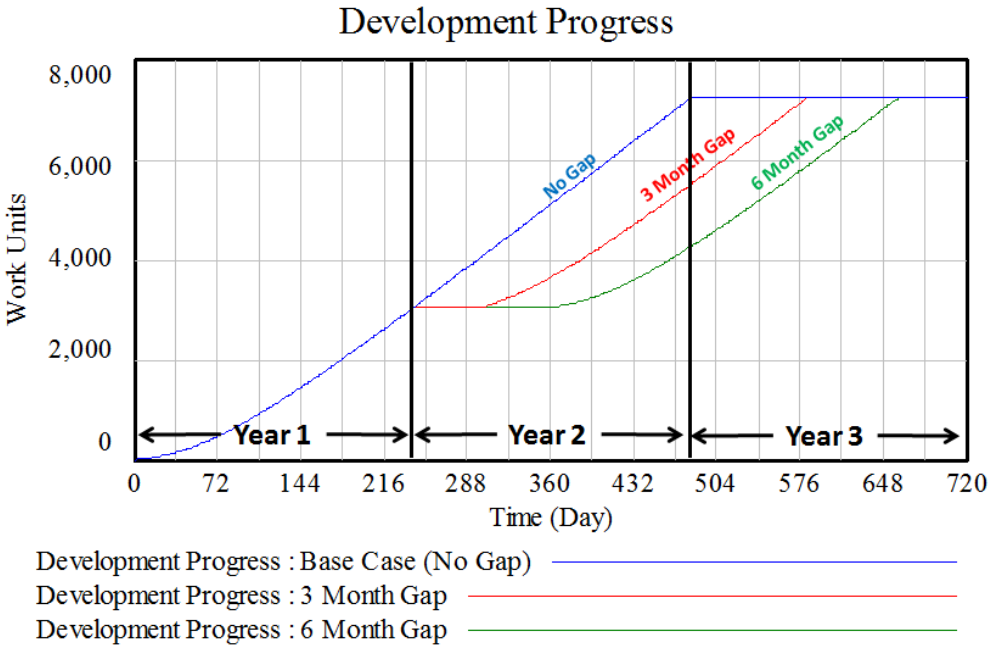


Figure 9. Affect of Funding Gap on Development Progress

Table 2. Affect of Funding Gap on Development Progress

Total Time to Complete Project Comparison						
Test Case	Day Complete	Total Delay (Days)	Total Project Delay (Calendar)	Gap Length (Days)	Gap-Adjusted Delay (Days)	Gap-Adjusted Project Delay (Gap Tax)
Base Case (No Gap)	480	N/A	N/A	N/A	N/A	N/A
3 Month Gap	581	101	5 Months	60	39	2 Months
6 Month Gap	660	180	9 Months	120	60	3 Months

Affect of Funding Reductions on Development Progress

A similar analysis can be conducted for the situation of a funding reduction in the second year – which, as explained earlier, is accomplished by reducing the workforce in the second year. Productivity of the base case, along with the cases of a 25% and 50% reduction in second year funding, are shown in Figure 10. The results are summarized in Table 3.

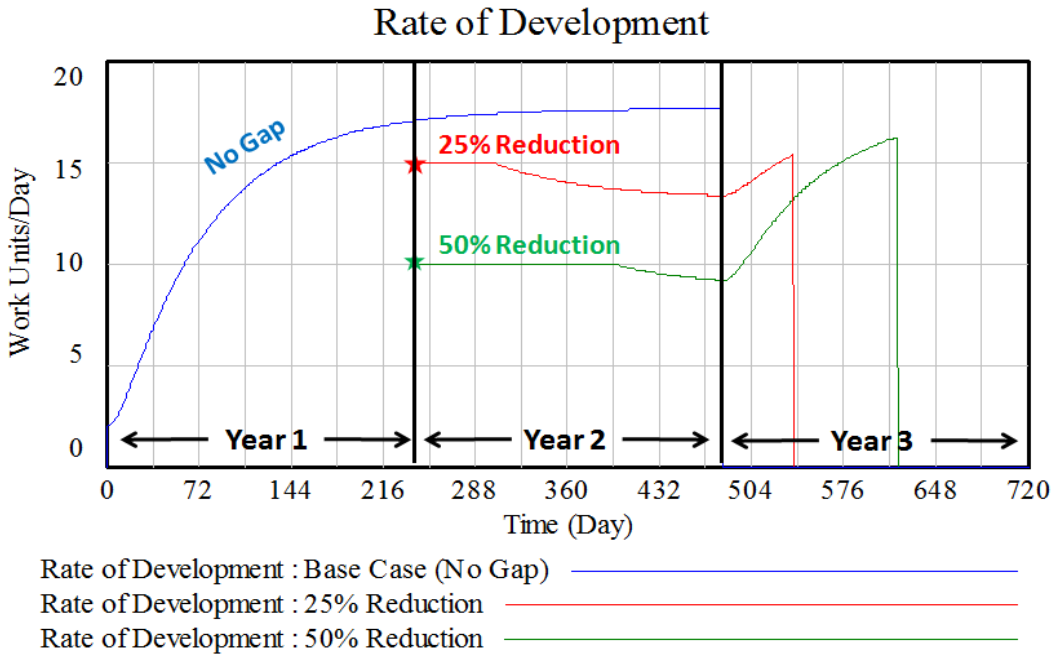


Figure 10. Affect of Funding Reductions Development Progress

Table 3. Affect of Funding Reductions Development Progress

Rate of Development Comparison (Work Units Per Day)		
Test Case	Following Funding Reduction	% Lost Productivity vs No Gap Full Funding
No Gap	17.0585	0%
25% Reduction	14.9777	12%
50% Reduction	9.98516	41%

The first significant difference to notice between Figures 8 and 10 is that there is much less drop in productivity in the case of funding reduction versus funding gap. That is because there is essentially no "gap tax," that is, no loss of knowledge since none of the staff that are continuing the work into the second year had to be put "on the bench."

Also, of note, in both the 25% and 50% funding reduction test cases, the loss in productivity experienced did not equal the percentage change in funding reduction. This occurs because New Staff are released first when funding is reduced. As mentioned previously, the New Staff are less effective and actually require a mentoring cost in the form of reduced production from Experienced Staff. Thus, by prioritizing the elimination of the New Staff actually somewhat increases the productivity of the Experienced Staff retained on the project, though there is a modest loss of overall productivity due to reduction of total workforce.

Affect of Funding Reductions of Development Progress

The reduction in funding for the second year, and the corresponding reduction in staffing and development productivity, does affect the rate of development progress and completion date, which like the funding gap, requires development efforts to continue into the third year, as shown in Figure 11.

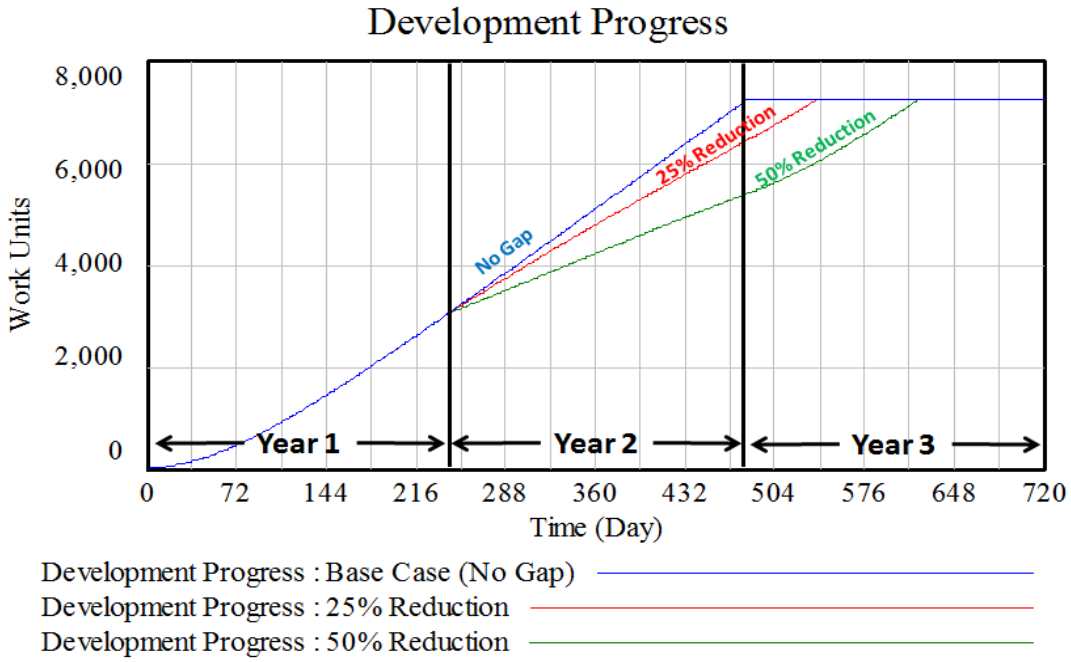


Figure 11. Affect of Funding Reductions of Development Progress

The extensions to completion times are summarized in Table 4. The results indicate a non linear relationship between the size of the funding reduction and the resulting delay in the project. Again, this is impacted by the prioritization of New Staff when releasing personnel. However, once the funding reduction reaches a certain level all and New Staff have been released and Experienced Staff must then be released. This explains the non linear relationship as funding reduction and delay increase mover closer to a one to one relationship.

Table 4. Affect of Funding Reductions of Development Progress

Comparison of Total Time to Complete Project			
Test Case	Day Complete	Total Delay (Days)	Total Delay Time (Calendar)
Base Case (No Gap)	480	N/A	N/A
25% Reduction in Year 2	536	56	3 Months
50% Reduction in Year 2	618	138	7 Months

Comparison of the Affect of Funding Gaps and Funding Reductions on Total Delay

The affect of funding gaps and funding reductions are compared in Table 5. These results suggests that, in general, funding reductions are cheaper and produce shorter delays. That is, given a choice between a funding gap and a funding reduction (of the same funding amount, i.e., 3 months and 25%, 6

months and 50%), it is better to have a funding reduction and keep a minimal amount of staff assigned to the project and keep work going in order to minimize the effect of the ramp up tax and the creation of a gap tax.

In particular, whereas a 3 month funding gap actually extends the completion date of the project by 5 months (the extra 2 months coming from the gap tax), a 25% reduction only extends the completion date by 3 months. This surprisingly good result comes from a short-term boost that the project gains by releasing the New Hires first. That enables the Experienced Staff to focus all their efforts on the development since there is less (or no) training of New Hires to be done. Of course, in a project of a much longer duration, there is a penalty to be paid for this boost since there are fewer trained Experienced Staff[13].

Table 5. Comparison of the Affect of Funding Gaps and Funding Reductions on Total Delay

Test Case (2nd Year Impact)	Total Delay (Days)	Total Delay (Calendar)
No Gap	0	0
3 Month Gap	101	5 Months
25% Reduction	56	3 Months
6 Month Gap	180	9 Months
50% Reduction	138	7 Months

Another interesting observation is that whereas there was no extension of the completion delay in the 25% reduction compared with a 3 month gap, the 50% reduction does result in a 7 month delay, which is more than the corresponding 6 month funding gap. This is primarily due to the limits on short-term boost that can be gained by releasing New Hires. Once all the New Hires have been released, (1) Experienced Staff must be released to get down to a 50% funding reduction, which significantly reduces productivity and (2) a much higher ramp up tax must be paid at the start of year 3 to get up to full staffing.

Key Insights

Placing employees on the bench is an effective way to reduce the ramp up tax and avoid the extensive security clearance delays that would result from trying to bring in new people from outside the agency. However, after a long period on the bench, the skills and knowledge levels have decreased to a point where they can be essentially classified as new staff to the project and must be fully assimilated back on to the project.

Another consequence of the process of keeping staff within the agency and the use of the bench is to increase the value of these cleared and badged personnel and thus increase their cost. Over time, this cost is likely to increase the overall project cost.

As mentioned above, it is interesting to note that a funding reduction in the second year does not produce an equal reduction in funding for the overall project. This is because releasing New Staff first has only a modest impact since (a) they were not yet fully productive and (b) they reduced the productivity of the Experience Staff that were helping them to assimilate onto the project. However, failing to retain and train New Staff may have negative effects on future projects but this effect was not investigated with this model.

[13] This illustrates other types of experiments that could be run using this model, such as projects of longer duration or the affect of a serious of related projects (e.g., multiple releases of the a system.)

A key finding is that, given a choice, a reduced funding level is preferable to a funding gap. During periods of reduced funding, a core group of experienced staff are retained and production continues at a reduced level.

Conclusions and Suggestions for Future Research

Budget fluctuations are a fact of life for anyone working in the government. The power of the purse is the primary method of control of government activities for the U.S. Congress. As noted earlier, there are many causes of budget fluctuations. The budget fluctuations that result are, however, not without cost. This research provides a starting point for investigating the effect of these fluctuations and methods for better understanding and attempting to minimize their cost.

The results match a general intuition that stopping and starting projects, i.e. creating funding gaps, is more costly than continuing work. But this paper makes the "intuition" more concrete. The existence of the ramp up tax, and consequent gap tax (essentially a re-ramp tax), provides the reasoning for this effect. Additionally, the effect of funding reductions can be minimized by the practice of releasing New Staff first. Again, this matches our intuition that any organization can, for a short time, focus all effort on production at the expense of training and maintenance. However, this will have negative long term consequences. Overall this research provides a base for future investigation of this topic. The logical next step will be to broaden the funding scenarios to better match reality. Additionally, more sophisticated production and development models should be developed to illustrate other organizational effects taking place. Finally, although this research was based on a detailed of study of a government agency, private sector organizations can also encounter similar "start and stop" situations in software development – and this research provides a basis for further study of those cases.

References

Abdel-Hamid, Tarek K, and Stuart E Madnick. (1991) *Software Project Dynamics: An Integrated Approach*. Englewood Cliffs, NJ: Prentice Hall, 1991.

Choucri, N. Electris, D., Goldsmith, D., Mistree, D., Madnick, S., Morrison, J.B., Siegel, M., Sweitzer-Hamilton, M. (2006), "Understanding & Modeling State Stability: Exploiting System Dynamics" (with N), *Proceedings of the 2006 IEEE Aerospace Conference*, Big Sky, Montana, March 5-12, 2006

Kadish, Ronald, Abbott, Gerald, Cappuccio, Frank, Hawley, Richard, Kern, Paul, and Kozlowski, Donald (2006) "Defense Acquisition Performance Assessment Report" https://acc.dau.mil/adl/en-US/18554/file/733/Defense%20Acquisition%20Performance%20Assessment%20Report%202006.pdf

Lieberman, Marvin, and David Montgomery. (1988) "First-Mover Advantages." *Strategic Management Journal,* 9, no. Summer Special Issue: 41–58.

Sterman, John, *Business Dynamics: Systems Thinking and Modeling for a Complex World*, McGraw-Hill/Irwin, 2000.

APPENDIX

Survey of DOD and Industry Experience with Net-Centric Systems

Allen Moulton, Stuart Madnick
MIT Sloan School of Management
Cambridge, MA 02139

1. Introduction

After the 1991 Gulf War and the end of the Cold War, the entire mission of defense was reevaluated and the size of the military establishment significantly cut back. A combination of experience in the Gulf War and the need to do more with less led to efforts to leverage information technology. One of the major directions was facilitating information sharing across different systems and organizational elements.

DOD is a large, complex organization with millions of both uniformed personnel and civilian employees, outsourcing to a large defense contractor community, as well as necessary information flows to and from other government agencies, and other national militaries and governments. Even after downsizing, the DOD budget is larger than most commercial enterprises in the world. The mission of DOD involves risks to life of warfighters and the security of the nation. Adversaries do not just compete, they shoot and blow things up. All this creates an urgency to effectively integrate necessary information despite the enormous complexity involved.

2. Data Standardization Efforts in the 1990s

From 1991 to 2003, DOD attempted to achieve information sharing using a department-wide data administration process (DoD Directive 8320.1, "DoD Data Administration," September 26, 1991). The objective was to catalog and standardize all data elements used anywhere across the entire defense enterprise. Rosenthal, Seligman and Renner (2004) reported that, after twelve years, out of an estimated million or more data elements only about 12,000 had been registered in the catalog – and hardly any had been reused. While success would have facilitated the flow of information across systems and organizations within the department, the standardization task was too complex to be achievable even with significant top level organizational commitment, large resources, and over a decade of effort. They conclude that "the process created a disincentive to agree on semantics, since agreement usually meant that somebody had to change implementation." (p.46)

In contrast to the failure of the department-wide data standardization effort, Rosenthal et al. also note that there were localized successes involving (1) small scope (less than 1000 tables), (2) a cohesive local enterprise with few or well-defined outside interfaces, and (3) a unified authority with effective control of requirements, funding, developers and users (p.46). Successful limited scope data interchange standards included the DOD meteorology data standard and the MITRE-developed Cursor on Target (CoT) schema. Successful data interchange standards involved factoring common information elements out of multiple systems, defining a standard representation for those elements that would be readily understood by all developers, and a loose-coupling strategy for implementation that would require software changes only at the interface.

CoT provides an example of the evolution of a narrow focus standard after its initial success. The first release of CoT provided What-When-Where information using only twelve required attributes. After CoT became widely used, Konstantopoulos and Johnston (2006, p.3) described CoT as very small, but "sufficient for about 90 percent of all information transfers needed in the Battle Management enterprise (blue force tracking, target deconfliction, strike warnings, ISR taskings, etc.)." In subsequent releases, CoT was elaborated into hierarchical structures to cover a wide range of information in greater detail. Based on the success of CoT, DOD developed UCORE (https://metadata.ces.mil/ucore/index.html) as an XML-based standard for sharing information which added the Who dimension, as well as a much richer description of What-When-Where compared to CoT. For example, CoT described Where as a simple regular cylinder located at a latitude, longitude, and height above a single ellipsoid model of the Earth. In UCORE, the same object at the same location can be described using the full power of GML (Geography Markup Language) which can represent complex polygons and a variety of coordinate systems as well as many related elements. As with many other standards, simplicity led to complexity and complexity led to different "standard" ways of representing the same thing.

3. ERP Systems in Industry and Defense

During the 1990s many firms in private industry began to introduce vendor-supplied packaged ERP systems to replace legacy transaction processing systems and databases for major segments of their business. An ERP system provides an integrated pre-built information technology solution for one or more functional areas across business units. In a survey of major firms that had adopted ERP, Ross (1999) found that the motivations were "(1) need for a common platform, (2) process improvement, (3) data visibility, (4) operating cost reductions, (5) increased customer responsiveness, and (6) improved strategic decision making." (p.2) ERP systems achieve these goals by replacing large numbers of separate legacy system silos with an integrated system and shared data. Ross reports that implementation cost ranged from $2 million to $130 million. The time from project initiation to initial operations ranged from one to five years. One the key factors was whether the firm decided to transform its business processes to match the assumptions in the ERP or customize the ERP to match the way the firm operated. While transforming business processes can lead to IT project success, it can also shift the balance of costs and benefits across the organization. For example, introduction of self-service travel expense reporting may reduce central administration cost at the expense of additional time required of employees and departments to enter and validate data and upload scanned documents. The willingness of an organization to accept business process change varies greatly, sometimes from normal bureaucratic inertia, but also because traditional processes may have evolved to be functionally adaptive to the business environment and business success. To be successful for the enterprise, any project must look at business value in addition to short term IT project goals.

Given the reported success in private industry, DOD looked to ERP systems to transform defense business systems for many of the same reasons identified by Ross. Programs were initiated in several functional areas of each of the services and by defense-wide agencies. According to GAO studies (e.g., GAO 2010a), all these efforts ran billions of dollars over budget and lagged many years behind schedule. Defense acquisition regulations impose delays and rigidities that private firms did not face and the complexity of defense business processes imposed daunting challenges. DOD also acquired multiple different versions of the same vendor

ERP system for different services or functional areas, resulting in a design that would have multiple larger silos in place of many smaller ones. Customization to meet defense needs introduced long delays and reduced the standardization benefit often seen in private industry.

4. Network-Centric Warfare

In their book *Netwok-Centric Warfare,* Alberts, Gartska, and Stein (1999) looked at private sector-driven growth in computing and communications technology and projected its impact on how future wars would be fought in what they describe as the coming Information Age. The 1990s witnessed unprecedented expansion in the commercial Internet and the advent of widespread use of computers and networks in households across the country. Reviewing commercial experience in applying evolving information technology to business operations and electronic commerce, they found four significant drivers: "(1) information technologies enable firms to create a high level of competitive awareness within their organizations and extended enterprises; (2) networking is enabling the creation of new types of information-based relationships with and among organizations that are able to leverage increased competitive awareness; (3) time is being compressed and, as a result, the tempo of operations is being increased; and (4) the cumulative impact of better information, better distribution, and new organizational behavior provides firms with the capability to create superior value propositions for their customers and dominate their competitive space." (p.51)

Alberts et al. conclude that the military must change to accommodate new realities in the Information Age. The key is "provision of nearly ubiquitous information services to all elements of the warfighting enterprise." (p.187) They describe the "Entry Fee" as network and other information technology infrastructure that will enable the organization to function in new ways. The constraints of "[l]egacy systems, designed as stovepipes optimized for one way of doing business, will need to give way to systems that are optimized to share and exchange information (with appropriate security)." (p.195) Information sharing is essential to achieving success (or surviving) on the battlefield of the future. But getting to a net-centric military will be a challenge: "While much has been learned about putting technology to use, the pace of technological advances has quickened to such a degree that current DoD methods of incorporating technology are well behind the power curve." (p.201)

One of the problems with keeping up with the rapid pace of technology change would be an acquisition process based on "a commonly held belief that most of the problems incurred in systems development could be traced to poorly articulated requirements, and if only the users would just do a better job writing requirements, everything would be fine." (p.203) Alberts et al. recommend the adoption of evolutionary acquisition (EA), originally proposed by AFCEA(1982), that

> "recognized that users, no matter how hard they tried, were unable to specify in advance all of their requirements. This inability was not found to be caused by a lack of effort devoted to the requirements process, as was previously thought. Instead it was the result of a faulty assumption. It was believed that users know what their requirements are, or at least *should* know. In fact, it is unreasonable to expect users to know, in any detail, what their requirements are or will be, when they do not have a full appreciation of the new or improved technologies, particularly in terms of implications for the environment or mission." (Alberts et al. pp.203-204)

Many of the EA proposals are similar to current concepts for “agile” acquisition. It has been exceedingly difficult to change the established acquisition processes to be more evolutionary or more agile.

5. Net-Centric Data Strategy

In 2003 DoD changed from the unsuccessful 1990s efforts to standardize a million data elements across the whole department to a new Net Centric Data Strategy (DOD 2003). DOD Directive 8320.02 (2004) established a new approach to facilitating data sharing based on agreements developed by communities of interest (COIs) which cross organizational boundaries to bring together providers and users of data about different topics (e.g., meteorology). The Net Centric Data goals were to make data:

- Visible – Descriptive information (metadata) about data assets are posted to a catalog visible and searchable throughout the Enterprise.
- Accessible – Users and applications post data to a "shared space" with data stored such that users and applications in the Enterprise can access it (except when limited by policy, regulation, or security).
- Interoperable – Many-to-many exchanges of data occur between systems, through interfaces that are sometimes predefined or sometimes unanticipated. Metadata is available to allow mediation or translation of data between interfaces, as needed.
- Understandable – Users and applications can comprehend the data, both structurally and semantically, and readily determine how the data may be used for their specific needs.
- Responsive to User Needs – Perspectives of users, whether data consumers or data producers, are incorporated into data approaches via continual feedback to ensure satisfaction.
- Trusted – Users and applications can determine and assess the authority of the source because the pedigree, security level, and access control level of each data asset is known and available.

The strategy was incorporated into the procedures governing the acquisition process through a Net Ready Key Performance Parameter (NR-KPP) defined by instruction CJCSI 6212.01. At each stage in the acquisition process programs would be evaluated on their compliance with the NR-KPP requirements. Since the NR-KPP was just one of many evaluation criteria, there was also the potential for waiving or relaxing the requirements. One unfortunate side effect of adding more formal evaluation procedures was to make the process more rigid and less “evolutionary” or “agile” as Alberts et al. had recommended. While the objective was to break down information silos and increase data sharing, the formal procedures may well have encouraged a focus on passing the hurdle rather than meeting the larger goals.

6. Early Net-Centric Success with Army Battle Command System

The Army’s rapid development of an integrated Army Battle Command System (ABCS 6.4) represented an early success with the net-centric concept. Prior to 2003 the Army had developed eleven Battlefield Functional Area (BFA) systems that brought information technology to command and control of ground forces. These BFA systems were developed

separately, each with an emphasis on delivering increasing capability in its functional area. Each BFA system had a separate acquisition program, with its own program manager, prime contractor, and budget. Systems were silos from the hardware up to user interfaces. There was limited data interchange among BFA systems, with the exception of some specially engineered interfaces for specific requirements to connect across silos (such as deconfliction between fire control and air traffic control). Different Army divisions implemented different combinations of BFA systems.

In 2003, Army Chief of Staff Gen. Schoomaker ordered a change to a new "Good Enough" strategy aimed at rapidly deploying a common integrated battle command system of systems (SoS) across the whole Army (Green & Mendoza 2005). A publish-and-subscribe hub was added to connect across the BFA silos. Each BFA system was modified to publish some of its data using an XML-based message standard developed within the ABCS communities of interest. BFA systems were also modified to be able to read data as needed from the hub. The development project was taken from concept to release into full system test in one year, with an additional year to test, train, and roll out of the system to divisions deploying overseas.

Greene and Greenberg (2010) reviewed lessons learned from the initial ABCS 6.4 effort and follow-on phases. The key factors they identified included: (1) strong leadership from the Army Chief of Staff and momentum gained from early success, (2) unified management of requirements portfolio and funding at the SoS level with sub-allocation to programs and integration activities, (3) logistical planning for field support and common systems for users to report problems and track fixes, (4) systems engineering, including integrated schedule planning and risk identification and mitigation, and (5) early and realistic testing both to identify and correct problems and to develop effective procedures for using and supporting the final product. They also noted that in this case software was a small part of the schedule, which was dominated by testing, training, and deployment.

7. Paying the "Entry Fee" – Developing Net-Centric Infrastructures

The Net-Centric "Entry Fee" identified by Alberts et al. is the underlying information technology infrastructure that would connect all suppliers and consumers of information. DOD named this infrastructure the Global Information Grid (GIG). The goal of the GIG was enabling "all DoD users (and their external mission partners) to find and share the information they need, when they need it, in a form they can understand, use, and act on with confidence; and protects information from those who should not have it." (DOD CIO 2007, p.7) Stove-pipe systems are classically built from the bottom hardware and network level all the way up to user-facing applications. The GIG concept would factor common low level elements out of individual system designs and implement them as shared services that could be used by all systems.

As shown in Table 1 many of the key GIG capabilities require a reliable and secure network that can reach from national and theater headquarters to the "tactical edge" where warfighters operate, as well as interconnection to other coalition and partner forces and agencies. Development of these network capabilities has proven challenging, particularly at the edge where bandwidth and environmental conditions are limiting factors (e.g., aboard smaller ships or with a special operations unit in the mountains). Over time, the term GIG has come to frequently refer to the network infrastructure more than the broad concept in the original vision.

Table 1. Target GIG Attributes
Department of Defense Global Information Grid Architectural Vision (June 2007) p.8, Figure 3.

Attribute	Description
Internet & World Wide Web Like	Adapting Internet & World Wide Web constructs & standards with enhancements for mobility, surety, and military unique features (e.g., precedence, preemption).
Secure & available information transport	Encryption initially for core transport backbone; goal is edge to edge; hardened against denial of service.
Information/Data Protection & Surety (built-in trust)	Producer/Publisher marks the info/data for classification and handling; and provides provisions for assuring authenticity, integrity, and non-repudiation.
Post in parallel	Producer/Publisher make info/data visible and accessible without delay so that users get info/data when and how needed (e.g., raw, analyzed, archived).
Smart pull (vice smart push)	Users can find and pull directly, subscribe or use value added services (e.g., discovery). User Defined Operational Picture v Common Operational Picture.
Information/Data centric	Info/Data separate from applications and services. Minimize need for special or proprietary software.
Shared Applications & Services	Users can pull multiple applications to access same data or choose same apps when they need to collaborate. Applications on "desktop" or as a service.
Trusted & Tailored Access	Access to the information transport, info/data, applications & services linked to user's role, identity & technical capability.
Quality of service	Tailored for information form: voice, still imagery, video/moving imagery, data, and collaboration.

The net-centric GIG architecture envisioned layers of shared enterprise services above the network level, such as messaging, collaboration, mediation, content discovery and delivery, user access (i.e., enterprise portal, as opposed to user access control), enterprise service management, people discovery, service discovery, service security and user authentication. These services could be developed and provided by any DOD element and made available to all through a service-oriented architecture (SOA) framework.

The Defense Information Services Agency (DISA) took the lead in moving forward with its Net-Centric Enterprise Services (NCES) program. NCES operates on DISA servers, offering a group of enterprise-level Web-based services as well as several user-facing applications. While originally conceived as the standard set of enterprise services for all of DOD, the program was reevaluated in 2008 and future development essentially halted with the initial set of services.

While NCES was under development, the services also began to implement core enterprise services to meet their needs (generally with the proviso that they would be supplanted by NCES when it became available, if it did what was needed). These development activities were wrapped into programs that also included new end user capabilities. In addition to DCGS, described below, these other enterprise services programs included:

- Army – Future Combat System (FCS) System of Systems Common Operating Environment (SOSCOE)
- Navy – Consolidated Afloat Networks and Enterprise Services (CANES)
- Marine Corps Enterprise Information Technology Service (MCEITS)

- Air Force – Global Combat Support System-Air Force (GCSS-AF)
- Air Force – Air Operations System-Weapons System (AOC-WS)
- DISA – Net-Enabled Command Capability (NECC)

Hardware and software platforms varied, depending on the requirements and preferences of the sponsoring service. These programs met different fates, some surviving or failing along with the end user functionality they were tied to. GCSS-AF was completed and deployed by the Air Force. The software system for the Army's SOSCOE was largely completed, but was orphaned when FCS was cancelled. DISA's NECC, never able to get services to agree on the semantics of command and control, was cancelled and funds redeployed to service C2 silos. The effort put into these programs was not entirely wasted, since there was substantial learning about what worked and what did not, but the anticipated benefits were only partially achieved.

8. Distributed Common Ground System

The Distributed Common Ground System (DCGS) is a family of systems for enabling shared access to intelligence, surveillance, and reconnaissance (ISR) information. DCGS evolved from an architectural concept proposed by the 1990s Defense Airborne Reconnaissance Office to rationalize the growing multiplicity of airborne reconnaissance systems, each of had been developed as a silo from ground station to airborne asset. Each of the services and the intelligence community launched its own program to develop its version of DCGS. As they evolved, these systems improved the integration of information to meet each service's particular needs, but the ability to share, integrate, and exploit the information has been less successful even as the volume of ISR information has multiplied. To break through the silo structure, DOD developed the DCGS Integration Backbone (DIB) which consists of a system of technology, standards, and protocols to facilitate interoperability. While bandwidth and network access is a critical problem in many locations, GAO (2010b) found that standardized meta-data tagging of information objects was the key limitation on finding and exploiting the vast amount of information that is being collected. While DCGS was initiated in 1998, GAO found that many of the systems have yet to reach full operational capability (although clearly they have been in heavy use in the wars in Iraq and Afghanistan).

Another issue is the integration of very large data objects, such as live video feeds, with very small data objects, such as human intelligence. For example, an operator watching a video feed may see a number of people coming out of a complex, getting into a number of vehicles, and driving off down a road. But the video alone is not sufficient to tell the operator whether the vehicle convoy is an insurgent groups heading out on a mission or a family going off to a wedding celebration. This type of differentiation requires on the ground tactical knowledge that may be derived from small fragments of human intelligence information that have been collected and analyzed.

Knowing that different types of information bear on the same problem is a difficult task when faced with time pressure and huge volumes of information. Current efforts on developing the Defense Intelligence Information Enterprise (DI2E) framework are addressing the gaps in information integration (Corrin, 2011).

9. Summary

A decade of effort toward achieving net-centricity has been frustrating. Programs have often developed systems that are more isolated as silos than net-centric. Few core enterprise services have been accepted or deployed widely. Programs have often been cancelled or stunted. Nevertheless, there has been progress, successes can act as a foundation for future development, and lessons can be learned from failures. The "how" of the process has proven as important as the "what."

Meeting these goals has proven difficult in the past and will likely continue to be difficult in the future. The software and the interaction of systems and people in the extended defense intelligence enterprise are inherently complex. In their study *Organizing for a Complex World*, Berteau, Ben-Ari, and Zlatnik (2009, p.2) conclude that "Complex defense and network-centric systems have proven to be difficult to develop on time and on budget, a consequence of the complexity inherent in both the systems and the acquisition environment." By modeling the process, we hope to be able to enable managers to experiment with alternative approaches to achieving the goals of information sharing in the context of defense intelligence.

As we found in our review of DOD experience and study of funding effects, the necessary regulations and constraints create significant hurdles in the path acquiring new systems. The Defense Science Board (2009) found that "**The conventional DOD acquisition process is too long and too cumbersome to fit the needs of the many systems that require continuous changes and upgrades**. Many existing programs are exceeding cost and schedule baselines, which cannot continue unabated." (p.4)

IT systems have historically proved even more difficult to acquire than weapons systems. In Section 704 of the FY10 Defense Authorization, Congress directed DOD to study the problem and recommend new approaches. The OSD Report to Congress (2010) proposed changes where "IT will be acquired as "time-boxed" projects delivering capability in an iterative fashion using mature technologies, while managed in capability-aligned portfolios to identify and eliminate redundancy." (p.2)

The same types of complexity and system acquisition problems also occur in industry. The National Research Council (2010) found that "[a]gile approaches and iterative, incremental approaches to software development have been receiving increased attention in industry in recent years and are having a significant impact on how software is developed and how systems are tested." (p.86)

A key question is how agile methods can be successfully applied within the context of DOD software acquisition which must follow laws and regulations governing, among other things, contracting procedures and system certification and test. However difficult the problem may be, it is not intractable and develop of System Dynamics models can assist managers in understanding, exploring, and choosing among potential courses of action within the constraints of regulations and the interactions of stakeholders.

References

Armed Forces Communications and Electronics Association (AFCEA 1982), *Command & Control (C2) Systems Acquisition Study Final Report*. Washington, DC: Defense Technical Information Center, I September 1982,

Alberts, D.S., Garstka, J.J., Stein, F.P., (1999) *Network Centric Warfare: Developing and Leveraging Information Superiority*, CCRP Publ., 2nd Edition (Revised). Aug 1999, Second Print Feb 2000.

Berteau, David J., Guy Ben-Ari, and Matthew Zlatnik (2009). *Organizing for a Complex World*. Center for Strategic and International Studies: Washington, DC.

Corrin, Amber (2011) .” DI2E framework seeks to unite wealth of intelligence data.” Government Computer News. (May 06, 2011) http://gcn.com/Articles/2011/05/03/Defense-IT-1-intelligence-enterprise.aspx?p=1

Defense Science Board. (2009) “Department of Defense Policies and Procedures for the Acquisition of Information Technology” Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics: Washington, DC, (March 2009).

DoD CIO (2007). “Department of Defense Global Information Grid Architectural Vision.” Version 1.0 (June 2007) http://www.msco.mil/documents/_7_GIG%20Architectural%20Vision%20-%20200706%20v1.0.pdf

DoD Directive 8320.1, “DoD Data Administration,” September 26, 1991.

DoD Directive 8320.02, "Data Sharing in a Net-Centric Department of Defense," December 2, 2004.

GAO (2010a) “DOD Business Transformation: Improved Management Oversight of Business System Modernization Efforts Needed”. U.S. Government Accountability Office, Report GAO-11-53 (October 2010).

GAO (2010b) “ISR: Overarching Guidance Is Needed to Advance Information Sharing”. U.S. Government Accountability Office, Report GAO-10-500T (Mar 17, 2010).

Greene, Harold J. and Greenberg, Janet (2010). “ABCS--Not Business as Usual.” *Defense AT&L*;May/Jun2010, Vol. 39 Issue 3, Special section p20

Greene, Harold J. and Mendoza, R. (2005) “Lessons Learned From Developing the ABCS 6.4 Solution.” *Defense AR Journal* (April-July 2005): 177–189. http://www.dau.mil/pubs/arq/2005arq/arq2005.asp.

Konstantopoulos, Dino and Jeffery Johnston. (2006) "Data Schemas for Net-Centric Situational Awareness" (presentation at the 2006 Command and Control Research and Technology Symposium, San Diego, CA, June 2006), http://www.dodccrp.org/events/2006_CCRTS/html/papers/073.pdf?q=cot.

National Research Council (2010) , “Achieving Effective Acquisition of Information Technology in the Department of Defense”. Committee on Improving Processes and Policies for the Acquisition and Test of Information Technologies in the Department of Defense; National Academies Press: Washington, DC.

OSD Report to Congress (2010). “A New Approach for Delivering Information Technology Capabilities in the Department of Defense.” (Nov 2010).

Rosenthal, Arnon, Len Seligman, and Scott Renner. (2004). “From semantic integration to semantics management: case studies and a way forward”. *SIGMOD Rec.* 33, 4 (December 2004), 44-50.

Ross, Jeanne W. (1999). “The ERP Revolution: Surviving vs. Thriving.” CISR WP 407, MIT Center for Information Systems Research: Cambridge, MA. (August, 1999).

UCORE Community Web-site https://metadata.ces.mil/ucore/index.html