The Role of Project Modularity in Information Systems Development

Completed Research Paper

Yi Nah Yeo
National University of Singapore
Singapore 117417
yinah.yeo@nus.edu.sg

Jungpil Hahn
National University of Singapore
Singapore 117417
jungpil@nus.edu.sg

Abstract

Recent surveys have shown that ISD project success rates are particularly low. Organizations are applying a wide array of information systems development methodologies (ISDMs) – both plan-based and agile, such as waterfall, scrum and XP – to improve information systems development (ISD) performance. However, as ISD projects often have different characteristics such as size, scope and complexity, prior studies have been focused on helping organizations better choose an ISDM for projects. Nevertheless, prior research has not taken into consideration the notion of problem modularity of ISD projects. In this study, we utilize the NK *fitness landscapes model to computationally examine via computer simulations the effects of problem modularity alongside various project environmental factors, and aim to answer the question: under various environmental factors, which ISDM should an ISD team adopt under various degree of problem modularity?*

Keywords: Information system development (ISD), information systems development methodologies (ISDM), ISD project complexity, problem modularity, computer simulation, agile development, waterfall development

Introduction

Modern organizations depend on information systems largely for enhancing efficiency and competitive advantage. This phenomenon is coupled with the prevalence of information systems development (ISD) efforts within organizations (Xia and Lee 2005). A fundamental goal of ISD projects is the delivery of high-quality systems that fits the needs and requirements of stakeholders. Studies have shown that one of the major challenges in ISD is the determination and delivery of system requirements, and this has led to the difficulty in reaching project success. For example, a recent report of the Standish Group (2009) on projects success rates shows that as much as 24% of all projects were complete failures (i.e., cancelled prior to completion or delivered but never used), 44% were challenged (i.e., delivered late, over budget, and/or with less than the required features and functions) and only 32% were considered successful (i.e., delivered on time, on budget, with required features and functions).

Since the 1980s, studies have shown that to ensure the smooth delivery of systems projects while containing risks and challenges, formal information systems development methodologies (ISDM) have to be in place as a form of structured development process (Brooks 1975). Hence, it is important to choose an appropriate ISDM that increases a project's success rate, as the use of an adequate methodology plays an important role in developing software, to assure that it is delivered within schedule, within cost, and meets stakeholder requirements and needs (Geambasu et al. 2011). Furthermore, choosing an inadequate ISDM, or an ISDM that does not suit the characteristics of a project (in terms of processes, resources allocated, complexity, scope and size) can hinder the success of the development project. However, the existing literature has only broadly investigated the factors (e.g., project size, complexity, and organizational structure) that affect the selection of ISDM, most of which are largely project managerial in nature (Modha et al. 1990). Referring to the field of product engineering and development, while it is important to look at the managerial aspects to choose an appropriate development framework, a product's architecture and design is also central to the selection of an appropriate development methodology to maximize development configuration effectiveness (Yan et al. 2007). Thus considering the area of ISD projects, it will be beneficial to look at the more structural aspect of ISD projects. In this study, we focus on one such structural property of ISD project, namely, the degree of requirements modularity. Studies into modularity have shown that by implementing formal methodologies that suits the modular nature of a project, further benefits can be made in terms of project quality and performance, and will be discussed further (Yan et al. 2007). Thus, examining project problem modularity can potentially provide us with insights to which ISDM might be more suitable for projects of various modularity levels.

Although the sequential waterfall methodology is still widely adopted by many organizations, practice has seen an evolution towards the agile/iterative methodologies, comprising of models such as Xtreme Programming, Crystal and Scrum. It is noted that the traditional sequential waterfall methodology can be used with success largely for developing systems for which the requirements are clearly defined from the beginning of the project (Geambasu et al. 2011), whereas agile methodologies develop software in an incremental manner in numerous iterations while adapting to requirements changes (Dybå and Dingsøyr 2008). Proponents of the traditional waterfall methodologies have argued that planning and designing is more straightforward as developers and users agree on system requirements early in the project, and makes progress more easily measured. It is also argued that systems can be designed completely and more carefully, based upon a more complete understanding of all system deliverables, reducing the likelihood of the "piecemeal effect."[1] Proponents have also criticized that the iterative nature of agile development may lead to an overall reduction in system quality as there is less emphasis on understanding the system as a whole early in the project, and this becomes more pronounced in larger-scale implementations or complex systems that include high levels of integration. In contrast, proponents of the agile methodologies argue that agile developments are more user-focused, with high quality software with better customer value due to the frequent requirements adaptations throughout the development process. Thus, proponents criticized the effectiveness of requirements and the possibility of delivering a dissatisfactory system due to the inflexibility to changes of the waterfall methodology. While there have

[1] The "piecemeal effect" is a development phenomenon that can occur as pieces of code are defined and subsequently added to an application where they may or may not fit well.

been some discussions about which methodology is better for which projects under conditions such as project size and complexity, level of system integration, problem modularity, and the extent of requirements changes (Mishra and Mishra 2011, Geambasu et al. 2011, Nejad and Irani 2012), there is a generally a lack of concrete and rigorous comparisons across methodologies. The arguments from proponents of both methodologies from practice present arguments without much principled justifications, thus it is difficult to determine which ISDM is superior under the various project conditions and the reasons behind the superiority. Similarly, academic research has also focused broadly on the selection of appropriate ISDM for projects based on various aspects of a project such as size, project complexity, development team and organizational structure (Vavpotič and Vasilecas 2012). However, despite 30 years of research on how to better choose an appropriate ISDM for projects, the same question still surfaces within the academic communities. As ISD projects are complex in nature, choosing an adequate and appropriate ISDM that will deliver the best value amongst the wide array of options is a difficult decision.

An important project characteristic that has yet to be examined by past research is *problem modularity* of ISD projects. As problem modularity contributes largely to project architecture, examining this particular aspect of a project may provide us with valuable insights into the problems of ISDM choice. Project modularity brings about vast advantages such as increased flexibility and rapid innovation, better ability to deal with complexity and the accommodation of future uncertainty. Modern organizations have since been embracing this "power of modularity" in product development projects (Baldwin and Clark 2004, Ulrich and Eppinger 1999). Furthermore, proponents of modularity have also described the use of modularity as a problem-solving strategy by making complexity manageable by enabling development at the level of modules, rather than the entire system, and in parallel (Baldwin and Clark 2001, Brusoni et.al. 2007). Similarly, the ISD process represents a problem-solving process of searching for configurations that add value to the project performance (Hahn and Lee 2010). Since both concepts are fundamentally conceptualized as problem-solving strategies, examining the problem modularity feature of ISD projects further may yield potential insights to choosing an appropriate ISDM for ISD projects for the enhancement of project success rates.

The notion of modularity is central in the design and production of software artifacts, especially for large and complex projects.[2] Modularity is a general set of design principles that involves breaking up the system into discrete chunks that communicate with each other through standardized interfaces or rules (Langlois 2002). Conceiving the design of a complex software artifact as a modular system means to apply the basic principle of "information hiding" that prescribes treating software modules as opaque entities. In essence, modularity aids in managing uncertainty and complexity (Kässi et al. 2008). In recent years, the widespread adoption of object oriented languages and the diffusion of component based development as well other popular trends in software engineering have affirmed at large this information hiding principle and the paradigm of modularity as common software and system development practices, aimed at speeding up the development process, increasing innovation and increasing the success rates and quality of ISD projects (Narduzzo and Rossi 2003, Boudreau et al. 2007).

Theoretically, software modularity affects software development and software quality, as the modular approach facilitates task decomposition, which is a strategy for efficiently organizing the work necessary to create deliverables. Thus, in theory, the degree of software modularity is expected to be positively related to software quality and thus, ISD project performance (Simon 1996, Conley and Sproull 2009). However, to reap the potential enhancement in ISD project performance brought about by modularity, it is essential to implement an appropriate formal development methodology that fits the modular nature of the ISD project. Studies into modularity have shown that by implementing formal methodologies that suits the modular nature of a project, further benefits can be made in terms of time, cost, quality and ultimately, performance. Thus, there is an explicit practical need to develop and implement appropriate methodologies that are able to guide the direction of the whole design and development process using modular concepts in a systematic manner (Yan et al. 2007, Duffy et al. 1998). For example, in the electronics industry, a study has shown that a difference in product quality and cost of up to 64 times is

[2] Modularity has been receiving an increasing amount of attention in a variety of fields. In recent years, modular approaches have been widely adopted by software engineering projects, and has been extensively discussed in a variety of literatures (Baldwin and Clark 2000, Brooks 1975, Narduzzo and Rossi 2003).

present when a development methodology that suits the modular practices of the project is implemented, as compared to the reliance on designers' natural inclinations (Yan et al. 2007, Duffy and Ferns 1998). We expect that the under-researched concept of modularity should also have a differential impact for different ISDM implemented for ISD projects.

To improve the success rates of ISD projects, it is thus beneficial to investigate the link between the two concepts, how different types of ISDM and various degrees of modularity can affect the effectiveness and success of an ISD project. This study aims to explore this linkage and its associated dynamic performance relationship. The main research focus of this paper is to investigate which form of ISDM (waterfall vs. agile) leads to better ISD project performance under various degrees of problem modularity. While it is important to select an adequate ISDM for the ISD project, is it also important to grasp the optimal degree and extent of modularity for the best project quality and performance, to aim for better ISD project success rates. However, it is also important to note that ISD projects exist in different forms – projects have variations in terms of size, scope, complexity and resources allocated to them. Since these project characteristics may affect the system development process, this study will take into consideration the impact of these project elements while examining the dual linkage between ISDM and modularity to enhance the robustness of the research.

Research Approach

This study adopts the computational modeling and simulation approach to a computational analysis of the research problem. Field studies such as surveys and case studies enjoy the luxury of realism. However, field studies may be limited in generalizability and in what can be observed and measured since it is difficult to manipulate all variables of interest in a field setting. Conversely, mathematical/analytical models allow formal analysis with rigor but must frequently rely on drastically simplified representations of organizations for analytical tractability and, as a result, cannot faithfully represent the richness of actual organizations. The simulation methodology frees manipulation of the project elements of interest in a formal model that incorporates a greater number of interdependent elements than is possible with a closed-form analytical approach, which helps acquire theoretical insights through various combinations of experimental conditions for greater generalizability (Amaral and Uzzi 2007, Davis et al. 2007).

Among the various simulation approaches, we employ the *NK* fitness landscapes model (Kauffman 1993, Kauffman and Weinberger 1989, Levinthal 1997) and extend it to ISD processes while considering the characteristics of modularity inherent in projects. The *NK* fitness landscapes model is an analytical framework for studying adaptive behaviors (Kauffman 1993). Its primary application is to statistically uncover the performance implications of the behavioral processes of goal-directed adaptive agents when faced with a variety of problem environments. As such, the researcher must specify two primary modeling constructs – 1) the problem space, and 2) the agent's behavioral rules for adaptation. The problem environment is abstracted and summarized into a fitness landscape (or performance landscape), whereas the agent's behavioral rules direct the agent's adaptation within the fitness landscape. The modeling framework provides researcher a means to systematically create stochastic fitness landscapes that encapsulate key environmental conditions (e.g., project size/scope, complexity, extent of problem modularity, etc.). Computational agents programmed with behavioral rules for adaptation (i.e., waterfall vs. agile ISDMs) are seeded onto these fitness landscapes and their adaptation behaviors are then observed and tracked. By simulating many different agents' behaviors within many different fitness landscapes, the researcher can uncover the statistical properties of the adaptation (e.g., average fitness level/performance at equilibrium, average time to reach equilibrium performance, etc.) in different problem contexts.[3]

In this research, we aim to understand the ISD process of projects with different modularity, thus we adopt a process-oriented view. ISD is conceptualized as a systemic work activity involving the process of system analysis, design and development (Korpela et al. 2002). This process can be conceptualized as a problem solving process where the agent (i.e., the ISD team) performs adaptive search strategies for an ISD project configuration that delivers value to the ISD project (Hahn et al. 2013). Specifically, the ISD team identifies a performance gap between the existing and desired states of the project configuration,

[3] Technical details of the basic *NK* fitness landscapes model are provided Kauffman (1993).

and carry out activities such as detailed user requirements gathering, generating alternative designs and solutions in the attempt to reduce the performance gap and achieve higher levels of ISD project performance. This process of search is adaptive and experiential. The ISD team is assumed not to have perfect knowledge of the true structure of the complexity of the problem space (for example, the interdependencies among decision variables), but the team is able to infer the value of different system designs when the specific configurations are considered (Cerveny et al. 1990).

A Model of ISD as Design Problem Solving

The model setup requires the specification of four features: (1) the representation of a project's performance (fitness) landscape; (2) the various modular design choices; (3) the process of the ISDM applications; and (4) the process of adaptation on the landscape.

Modeling the Project's Performance Landscape

A project p, is represented by a set of N decision variables, $p=\{d_1, d_2, \ldots, d_N\}$, where each decision d_i can take on either one of the two possible values (0, 1). For example, in the case of the development of a sales and marketing system, d_1 can be the decision to enable customer order tracking (d_1=1) or not (d_1=0), d_2 can be the decision to enable the sending of push notifications (d_2=1) or not (d_2=0), so on and so forth. Each decision contributes to overall value of the project and the value contribution of each decision depends not only on the choice made concerning that decision (i.e., d_i=0 or 1) but also on choices regarding K other decisions. In other words, each decision variable may be tightly linked to other decision variables. Thus, the performance of the project depends on the performance contributions of all decision variables based on the interdependencies among them.

In the most extreme cases, where there are no interdependencies among the decision variables, and all decision variables are independent (i.e., the choice of altering one decision will not affect the performance contribution of other decisions), each decision then contributes independently to the project performance. For example, the decision to enable customer order tracking or not does not depend on the decision of whether to enable the sending of push notifications. This results in a performance landscape that is smooth with a single peak. On the other hand, where there are strong linkages amongst decision variables, then the project performance contributions of all decisions become interdependent – choice of altering one decision will affect the performance of other related decisions. This results in a performance landscape that is rugged and with multiple peaks. For example, the decision to enable customer order tracking will depend on the decision to enable the access of customer accounts by the customer himself. The extent of interdependencies ranges from K=0 (i.e., most smooth) to K=N–1 (i.e., most rugged), with intermediate values of K representing increasing levels of project complexity. The patterns of interdependencies among design variables can be defined and represented in an $N \times N$ influence matrix (**INF**), where INF_{ij} = 1 if (column) decision variable j influences the value contribution of (row) decision variable i, or 0 otherwise. The influence matrix allows the modeler to specify the complex nature and modular structure of the ISD project requirements.

More formally, the performance contribution (c_i) of each decision variable (d_i) is dependent on its own value as well as the setting of K other interdependent decision variables, where $c_i = c_i(d_i|K \text{ other } d_j\text{'s})$. Without loss of generality, we assume that all decisions carry equal weights of contributing to the overall performance of the project, thus the average of all contributions of decision variables is used as a measure of overall project performance outcome.

Modeling the Design Structures

To model modularity, we arrange the interdependencies of the decision variables into clusters. Each cluster in the landscape represents a module. To examine the effects of various ISDM with respect to project modularity, we consider three design structures of varying degrees of problem modularity – perfectly modular, imperfectly modular, and non-modular/random (Ethiraj et al. 2008). Each design structure comprises of the total number of decision variables N, the number of modules M, and the total number of interdependencies R (i.e., $R = NK$). All 3 design structures will have the same value of R (i.e., the same number of interdependencies spread across the various decision variables). However, the distributions of the interdependencies are different for each design structure, which determines the

degree of modularity. For each cell INF_{ij}, a value of 1 represents the dependency between the decision variables d_i and d_j (i.e., the performance contribution of d_i depends on the value of d_j). In the perfectly modular problem structure, all interdependencies are within module (see Figure 1a). In the imperfectly modular structure, most interdependencies are within modules whereas there exists some interdependencies across modules (see Figure 1b). In the non-modular/random structure, there is an equal probability of having interdependencies within and across modules (see Figure 1c).

Figure 1. Varying Degrees of Problem Modularity (*N*=12, *M*=3, *K*=4)

a) perfectly modular

$$\begin{pmatrix}
1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1
\end{pmatrix}$$

b) imperfectly modular

$$\begin{pmatrix}
1&1&0&1&0&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
1&0&1&1&0&0&0&0&1&0&0&0\\
1&1&0&1&0&0&1&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&1&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&1&0&0&0&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&1&0&0&0&0&0&1&1&1&0\\
0&0&0&0&0&0&0&0&1&1&1&1
\end{pmatrix}$$

c) non-modular/random

$$\begin{pmatrix}
1&1&0&0&0&1&0&1&0&0&0&0\\
0&1&1&1&0&0&0&0&1&0&0&0\\
1&1&1&0&0&0&0&1&0&0&0&0\\
0&1&0&1&1&0&0&0&0&1&0&0\\
0&0&0&0&1&0&0&1&0&0&1&1\\
1&0&0&0&0&1&0&1&0&0&1&0\\
0&0&1&0&0&1&1&0&0&1&0&0\\
0&1&0&0&0&0&1&1&0&0&0&1\\
1&0&1&0&0&1&0&0&1&0&0&0\\
1&0&0&0&0&0&0&1&0&1&1&0\\
0&0&0&0&1&1&0&0&0&0&1&1\\
1&0&0&1&0&0&0&0&1&0&0&1
\end{pmatrix}$$

Notes: *N*=12, *K*=4 and *M*=3. The modules are highlighted with the dashed boxes.

Modeling the ISD Process

The agents' (ISD teams') adaptive behaviors are modeled as incremental experiential search. For the agile development process, the scope of search is within the module currently being attended to, whereas for the waterfall development process, the scope of search is the entire system. The agents perform local search at the module-level, attempting to enhance the performance at module-level. In each simulated time period, the agents select a neighboring decision configuration at random (out of the decision variables that are within scope), and evaluate the performance implications of the new configuration. If the new configuration results in a performance increase, the configuration is adopted and implemented; else, it will be discarded.

More specifically, the *agile* development process proceeds through I iterations (equal to the number of modules, $M>1$) where each module becomes the scope of search within iteration. For example, the ISD team will consider decision variables d_1 through d_4 during the first iteration (see Figure 1). Each iteration ends when the agent can no longer find higher performing design configurations within the focal scope. Upon completion of an iteration, the agent may consider changes to modules previously completed during prior iterations. This return to prior modules represent the notion of refactoring in agile development. The waterfall development process is thus a special case of the agile development process where there is 1 module (M=1) comprising of all decisions (i=1,…,N) and 1 iteration (I=1).

Modeling the True Underlying Design Structure

Considering the modular design structures (i.e., the perfectly and imperfectly modular structures), by varying the value of the number of modules M, we can examine the effects of over-modularity and under-modularity. This notion is based on the basis that agents do not have a-priori knowledge of the true design structure of the problem (i.e., they do not have the knowledge of the optimal number of modules to have within a project). Thus, their iteration plan can be classified as over-modularized, under-modularized, of adequately modularized if the plan fits the true underlying design structure. Letting the true design structure be M_1, a design structure is considered to be over-modularized when $M_1>M$, and under-modularized when $M_1<M$. Figure 2b shows the case of adequate modularization which fits the true underlying modular structure of the project (M_1=M=3), Figure 2a shows the under-modularized structure ($M_1<M$=4) and Figure 2c shows the over-modularized structure ($M_1>M$=2).

Figure 2. Over-, Under- and Adequate Modularization

a) under-modularized (M=4)

$$\begin{pmatrix}
1&1&0&1&0&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
1&0&1&1&0&0&0&0&1&0&0&0\\
1&1&0&1&0&0&1&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&1&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&1&0&0&0&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&1&0&0&0&0&0&1&1&1&0\\
0&0&0&0&0&0&0&0&1&1&1&1
\end{pmatrix}$$

b) adequately modularized (M=3)

$$\begin{pmatrix}
1&1&0&1&0&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
1&0&1&1&0&0&0&0&1&0&0&0\\
1&1&0&1&0&0&1&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&1&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&1&0&0&0&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&1&0&0&0&0&0&1&1&1&0\\
0&0&0&0&0&0&0&0&1&1&1&1
\end{pmatrix}$$

c) over-modularized (M=2)

$$\begin{pmatrix}
1&1&0&1&0&1&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0\\
1&0&1&1&0&0&0&0&1&0&0&0\\
1&1&0&1&0&0&1&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&1&0\\
0&0&0&0&1&1&1&1&0&0&0&0\\
0&1&0&0&0&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&1&1&1&1\\
0&0&1&0&0&0&0&0&1&1&1&0\\
0&0&0&0&0&0&0&0&1&1&1&1
\end{pmatrix}$$

Notes: N=12, K=4 and M_1=3. The modules are highlighted with the dashed boxes.

Validation of the Computational Simulation Model

An important step in the simulation-based theory development process is the validation of the computational model (Davis et al. 2007). However, the notion of validity for computational models is judged with a different set of criteria as for empirical models in field studies due to the high level of abstraction of computational models.[4] Of particular relevance is the assessment of content and construct validity. First, the assessment of content validity should answer the question of "does the model make sense to a group of experts". In other words, the development of the computational model must persuasively argue to a relevant group of colleagues that the model captures some important aspect of the phenomenon, and as a result provoke an expectation to learn something about the phenomenon from the model. We believe that our model construction efforts and resulting discussion of the model is convincingly portraying the ISD process as design problem solving and that the modeling framework of *NK* landscape models is appropriate insofar as it enables us to capture the important and salient aspects of the phenomenon we are investigating (i.e., task partitioning, design choice interdependencies, complexity etc.). Second, the assessment of construct validity must ensure that the model and the computation contain parameters, variables and relations which yield outcomes with correspondence to the real world. To this end, we validated our *NK* model by verifying that our (basic) results are consistent with findings from prior research or our intuition. For example, the results show that (1) as ISD complexity increases, ISD performance decreases and (2) the fit between the true underlying structure of the problem and the problem modularization produces the best performance, which are consistent with prior research findings.

Results

In order to examine the effects of degrees of modularity across the two different kinds of development methodologies on ISD project performance, we conduct two levels of simulation experiments. The first level (i.e., baseline) examines the basic relationship between the types of methodologies and degrees of modularity. The second level manipulates one parameter at a time – i.e., the number of modules (M), the problem complexity (K), size and scope, as well as resource availability for the ISD project.

Monte Carlo techniques are used to minimize any spurious effects due to initial settings. All results are based on 50 independently generated fitness landscapes for each influence matrix (representing the

[4] Burton and Obel (1995) provide an excellent discussion of validity of computational models, where they note that "[A] valid computational model is one that is effectively appropriate to the end goal" (p. 58). In other words, relevancy for the intended purpose must be the focus of validation. Our *NK* model of ISD is an *illustrative / intellective study of quasi-realistic organizations*, where the purpose is "… to explore the implications of reasonable assumptions about organizational behavior, in order to determine what the world is like when these assumptions are true …" (Cohen and Cyert 1965), that is, to illustrate in a quantitative form general hypotheses about some phenomenon and derive implications.

different design structures), with 20 ISD projects randomly seeded onto each fitness landscape. The average performances across agents are computed. The performance of the ISD project is measured as a portion of the highest performance attainable on each landscape.[5]

Baseline Performance Relationship Between ISDM and Modularity

The first experiment examines the basic performance relationship solely between the software development methodologies and the degrees of modularity (while holding other factors constant). We initially set N=16, K=6 and M=4.[6] The analyses of the basic performance relationship between the software development methodologies and the degrees of modularity are based on 14 experimental configurations of the ISD project – (1 perfectly modular structure + 3 imperfectly modular structures + 3 non-modular/random structures) × 2 ISDM applications = 14 experimental conditions.

Figures 3a and 3b shows the performances over time for the agile and waterfall ISDMs on projects with 3 degrees of problem modularity. The results suggest that when the agile methodology is adopted, in terms of speed, perfectly modular is the first to plateau, followed by the imperfect and non-modular structures. In addition, the imperfect modular structure performs best in terms of peak performance. However, the peak performance for perfectly modular structure is significantly lower than both the imperfect modular and non-modular design structures when agile methodology is applied. Thus, although the perfectly modular structure has a faster initial performance improvement, the end performance attained is less satisfactory than the other designs. Conversely, when the waterfall methodology is adopted, performance is relatively consistent for across imperfectly modular and non-modular structures. In contrast, the peak performance for perfectly modular structure is relatively lower than the other two structures. The performances of all three design structures take similar time to plateau. Figure 4 shows that the agile methodology performs better than waterfall methodology in terms of peak performance attainable for both the imperfect and non-modular structures. However, peak performance attainable for perfectly modular structure is consistent for both the agile and waterfall methodologies.

When a project is modularized, the adaptive and iterative nature of the agile methodology aids in producing projects of higher quality performance, which outperforms that of waterfall. However, when interdependencies between modules are absent, a modularized project structure seems to impede the performance of the project, regardless of the ISDM used. As an adaptive development approach, agile operates in modules, allowing for change in requirements and adaptations of decisions. Perhaps due to the adaptive nature of agile, its benefits are not largely realized in the perfectly modular structure, as the landscape is smooth and there are zero interdependencies between modules. Thus, a certain extent of interdependencies between modules in a project has to be present for agile to realize its benefits.

---

[5] In the *NK* landscapes simulations, the parameter K (representing the overall complexity of the landscape) determines the maximum fitness value within the landscape. Smooth landscapes (small K) generally have smaller maximum fitness value that are easier to attain, as compared to rugged landscapes (large K) which have larger maximum fitness values that are tougher to reach. Thus, when comparing performance across complexity levels (i.e., across K), it is appropriate to compare fitness values that are normalized to the maximum attainable performance levels rather than the raw fitness measures (Hahn and Lee 2010).

[6] When N = 16, there are 2^{16} = 65,536 possible configurations of ISD projects, which represents a sufficient level for project size as an exhaustive search through all 2^{16} IS configurations is impractical. Later, we relax this assumption and test the robustness of our results by running the simulation experiments with N = 20 (or 1,048,576 possible configurations).

Figure 3. Performance over Time - ISDM × Degrees of Modularity

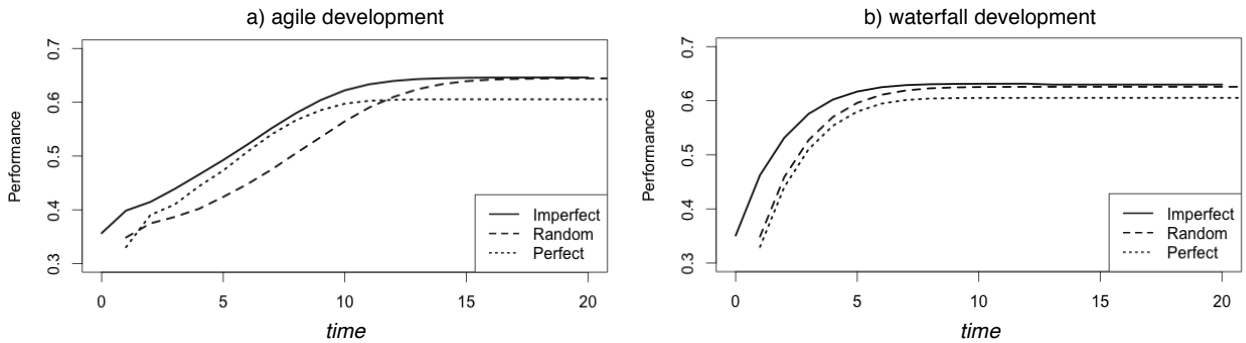
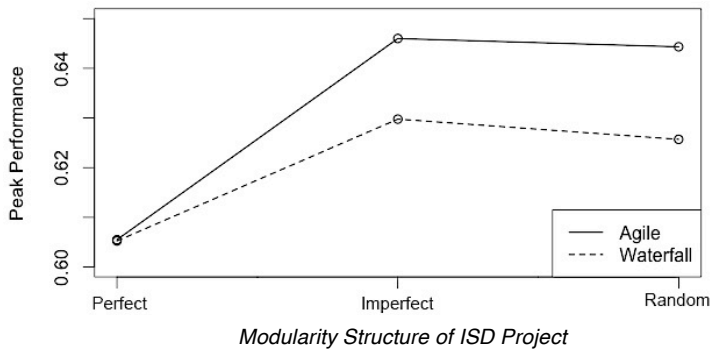


Figure 4. Final Performance - ISDM × Degrees of Modularity

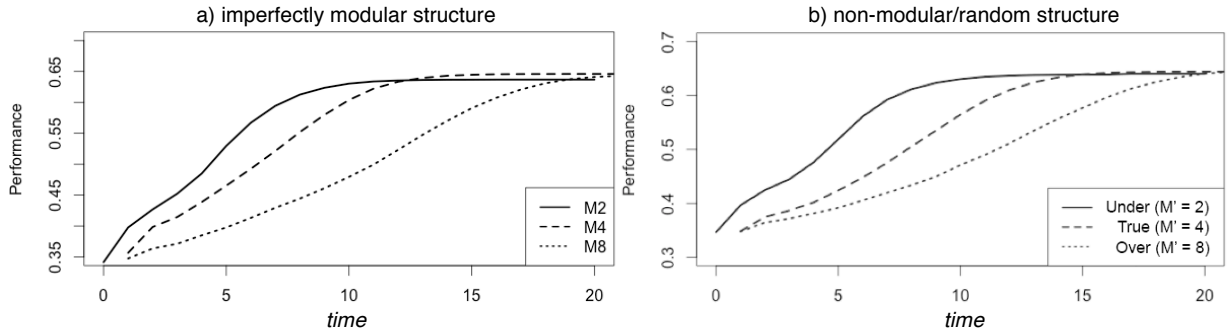


Effects of Under- and Over-Modularization

Simulation experiments were conducted on the imperfectly modular and non-modular design structures to examine the effects of under- and over-modularization. Here, we maintain the project size and complexity at N=16, K=6, set the true underlying structure to be M_1=4, and vary the number of modules M planned for the agile development. We set M=2 and 8 for under- and over-modularization, respectively. Here, since modularization is only relevant for agile development (i.e., for waterfall development, the number of modules is always one; M=1)

The results, summarized in Figures 5a, suggest that when the project structure is imperfectly modular, performance for the true underlying structure (i.e., M=4) reaches the highest level of performance as compared to the under- and over- modularized structures, highlighting the importance of iteration planning. In particular, the under-modularized project yields a lower peak performance, and the over-modularized project takes a significantly longer time to peak, suggesting that when an ISD project is planned in an over-modularized manner, there are too few decision variables to be considered in each iteration, and as a result performance improvements slow down and the benefits of adaptability are diminished. When the project structure is non-modular, performance peaks are comparable across all levels of modularization (M=2, 4 and 8). However, performance plateaus fastest when project structure is under-modularized and slowest when over-modularized. These results suggest that iteration planning is important when there exists some inherent level of problem modularity in the project (i.e., imperfectly modular structure). However, when such underlying structures are not present (i.e., non-modular/ random structure), iteration planning is no longer critical.

Figure 5. Performance over Time – Modularization

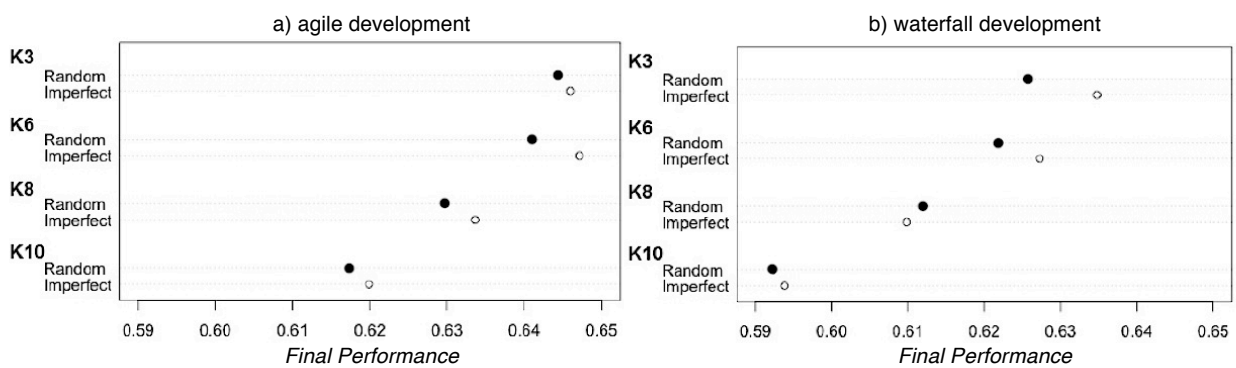


Effects of Project Complexity

We conducted additional experiments to take into account project complexity while examining the performance relationship between the development methodologies and degrees of modularity. While we vary the design complexity by altering the distribution of the interdependencies, we vary the cognitive complexity by altering the environmental complexity of the landscape within the parameter K of the NK model. Hence, in this experiment, we again set N=16 and M=4 as before, but vary the cognitive complexity with K={3, 6, 8 and 10}.

Figures 6a and 6b summarize the results. The results show as complexity increases, peak performances decrease (as can be expected) for both the imperfect modular and non-modular design structures, in both cases where the agile and waterfall methodologies are applied respectively. Further, when agile development is applied, projects with an imperfect-modular design structure outperform that of a non-modular structure. In contrast, when waterfall is used, the performance gaps between imperfect-modular and non-modular close in as complexity reaches higher values (i.e., K=8 and 10).

Figure 6. Final Performance – ISDM × Degrees of Modularity × Project Complexity



Effects of Project Size

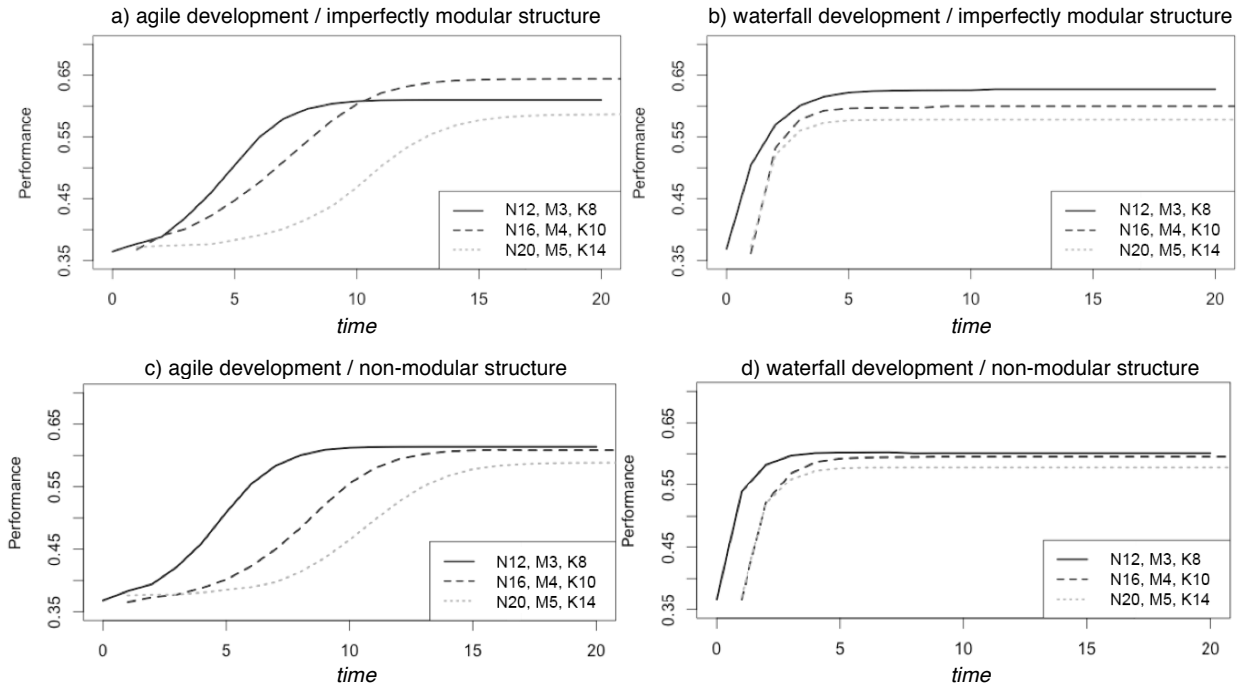
To examine the impact of project size, we conducted a series of experiments that varies project size (via parameter N of the NK model) and scope while examining the relationship between ISDM and problem modularity. Here, we varied the number of decision variables N={12, 16, 20} and adjust the overall complexity K and the number of modules M accordingly (i.e., K={8, 10, 14} and M={3, 4, 5} when N={12, 16. 20}, respectively).

Figures 7a through 7d summarize the results. We observe that when the agile methodology is applied in the imperfectly modular design structures (see Figures 7a and 7b), a project of smaller size (i.e., N=12) results in a much lower performance than projects of larger size and scope (i.e., N=16 or 20). In contrast, when the waterfall methodology is applied, peak performance decreases as project size and scope increases, uniformly. In addition, the results suggest that the agile methodology outperforms waterfall with smaller projects (i.e., N=12 or 16), but yields similar performance levels as waterfall with large projects (i.e., N=20).

It seems that while agile methodology works better with modularity than waterfall, these benefits are realized only when the project has a decent size and scope. When the size and scope of a project falls below a threshold, the benefits of agile are diminished. However, when the project size and scope becomes exceedingly large, the comparative benefits of agile over waterfall is diminished. Predictability of a project increases when a project has a smaller size and scope, and decreases when project size and scope is large. As a predictive methodology, waterfall development falls in performance when project size increases.

With non-modular design problems (see Figures 7c and 7d), as project size increases, the resulting performance decreases, consistently for both agile and waterfall development. Again, this suggests that the adaptive benefits of agile development are only realized when there exists some structural modularity in the ISD projects.

Figure 7. Performance over Time – Project Size



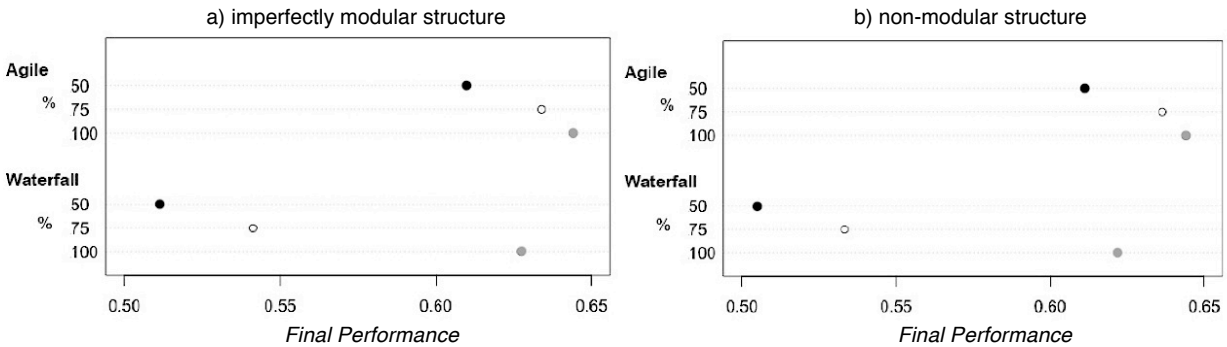
Effects of Resource Availability

This set of simulation experiment considers the effect of the project resource availability while examining the relationship between ISDM and modularity. With limited resources (cost, time and effort), the extent of the search of improvement of the project may be impacted. We create resource constraints by limiting the number of neighboring decision variables to be considered (search scope=50%, 75% and 100% of all possible neighboring configuration) for local incremental improvement. In other words, due to cost, time and/or HR constraints, the ISD team cannot perform exhaustive search of alternative configurations but resorts to investigating a subset of them. All other factors (size, complexity and number of modules for agile development) are held constant at N=16, K=6 and M=4.

Figures 8a and 8b summarize the results. It can be noted that with resource constraints, the peak performance of waterfall falls heavily below that of agile development in both the imperfectly modular and non-modular design structures. When the agile methodology is applied, peak performance decreases slightly as resources availability are constrained. With waterfall development, however, ISD performance is heavily dependent on the resources that can be deployed – even small constraints (e.g., reducing the scope of search by 25% leads to significant decreases in performance). Another interesting observation here is that the results are similar across imperfectly modular and non-modular design structures, suggesting that the detrimental impacts of resource constraints apply for all types of ISD projects regardless of their structural properties.

When there are resource constraints, cost, time and effort budget are cut. In the case of waterfall methodology, when there are changes that need to be made later in the project, these changes may not be able to be realized due to resources constraints, as such changes in a waterfall environment are often costly and time-consuming. In contrast, agile promotes flexibility and adaptability throughout the course of the entire project due to modularization. As such, changes are often not as costly as that of waterfall, in terms of cost, time and effort. As such, when resource constraints are high, the impact on the performance of the waterfall methodology is much heavier than that of agile.

Figure 8. Final Performance – ISDM × Degrees of Modularity × Resource Constraints



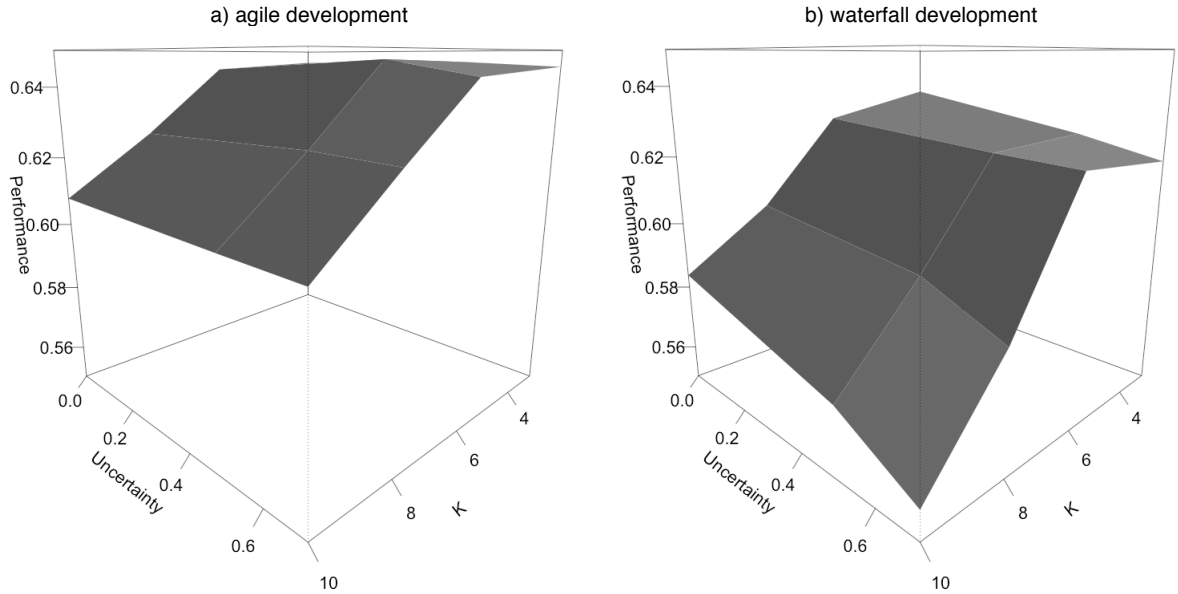
Effects of Project Uncertainty

In this experiment, we introduce the notion of requirements uncertainty into the model. Uncertainty can be conceptualized and modeled as the imperfect assessment of the implications of making design choices – the higher the uncertainty, the higher possibility of inaccurate assessment of the performance implications of selecting an alternative project configuration (Hahn et al. 2013). That said, the extent uncertainty will decrease overtime throughout the course of the ISD project as more feedback is obtained from the users at the end of each iteration. As the project nears completion, uncertainty will approach zero.

The results for the imperfect-modular case (see Figure 9a) suggest that when agile development is applied, as complexity and uncertainty increases by a small amount, performances for all cases are not impacted significantly. However, performance is adversely impacted as complexity increases beyond a certain threshold (i.e., $K>4$). We also note that project uncertainty does not impact agile performance significantly. For agile development, project complexity is much more influential to performance than project uncertainty. Conversely, when waterfall development is applied (see Figure 9b), performance drops in a steep manner when either project complexity and/or complex uncertainty increases. In essence, the agile methodology suffers when complexity is high, but is resilient to uncertainty, whereas the waterfall methodology is negatively impacted by both parameters. We experimented with the non-modular case as well (results not shown here due to space limitations). The results show that quantitatively, the performance of all non-modular settings were lower than that of the imperfect-modular cases. Similar qualitative results to the imperfect-modular design structures were obtained with non-modular design structures.

When project complexity and uncertainty are high, the requirements and technologies involved are complex, thus predictability inevitably is low. Due to the predictive nature of waterfall methodology, and also the inability to refactor and make significant changes to the earlier parts of the project, the waterfall methodology yields a much lower performance when project complexity and uncertainty increase. In contrast, the adaptive nature of agile methodology allows it to adapt to changing requirements and other project elements through adaptation and refactoring in each iteration throughout the project development. Thus, when complexity and uncertainty are high, agile performs better than waterfall, despite the possibility of taking a longer time to reach its peak performance.

Figure 9. Final Performance – ISDM × Project Complexity × Project Uncertainty



Discussion and Conclusions

Extant research on the selection of ISDM for projects have largely revolved around project characteristics such as project size, complexity and organizational structure. Minimal emphasis has been given to the problem modularity of projects. Using the *NK* fitness landscape model as an analytical framework, we computationally explore the impact of various levels of modularity on the application of agile and waterfall methodologies. We also examined the impact of project elements such as complexity, uncertainty, size and scope, as well as resource availability on the project performance with various degrees of modularity. We aimed to provide some insights into the question: Under various environmental factors, which ISDM should an ISD team adopt under which degree of problem modularity?

Our simulation experiments yielded several important insights. When the agile methodology is used, the imperfect modular structure performs better than that of the non-modular structure in all cases. However, the perfectly modular structure underperforms significantly. When the waterfall methodology is used, the performances of the imperfect and non-modular structures are relatively consistent, with the perfectly modular structure underperforming. Further, the effects of under and over modularization are only apparent in the projects adopting the agile methodology, where both under and over modularization are seen to deal negative effects on performance. The effect of increasing complexity as well as uncertainty decreases performances of projects adopting both the agile and waterfall methodology, with greater impact on waterfall projects. On the other hand, the effect of increasing project size decreases waterfall project performance consistently, whereas agile projects of relatively small and relatively large sizes underperform. When resource availability is limited, the impact on the performances of waterfall projects is significantly greater than that of agile projects. In all cases, the imperfectly modular projects are more resilient than non-modular projects when the agile methodology is used.

Implications of Results

To answer the question that we posed, when a project problem is adequately modular, of a decent size and scope, the project team can attain the highest possible performance using the agile methodology. However, as discussed above, a fully (i.e., perfectly) modular problem structure and a small project scope will instead diminish the benefits of agile. This observation may seem counterintuitive – some studies have arrived at notions where agile methodologies are better suited for smaller projects due to the adaptive and flexible nature (Awad 2005). Agile methodologies promote the ideas of collaboration,

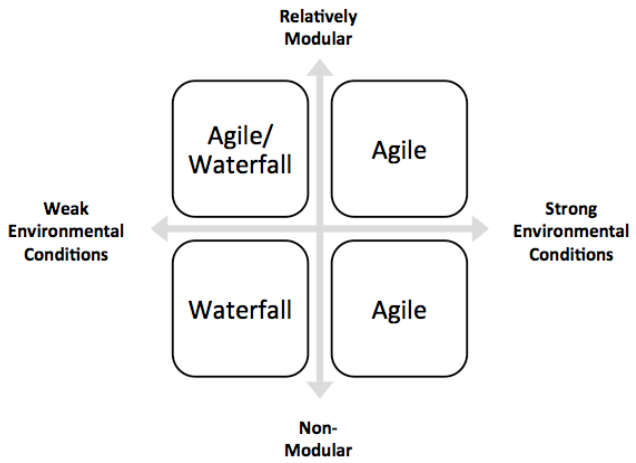
communication and iterations, which aim to encourage creativity and innovation. However, when an ISD project is of a very small scope and size, the problem becomes much simpler. Intuitively, the ISD team need not consist of a large pool of human capital as well. In this case, it becomes tougher to harvest the benefits of agile. In contrast, a certain extent of problem modularity – a level between being perfectly modular and non-modular – brings about major benefits of the agile methodology, for supporting adaptability and flexibility. Improvements can be made to modules developed before, to achieve an overall higher quality.

In addition, project teams must also be cautious as to whether or not the ISD project plan is under or over modularized. When a project is under-modularized, the number of decision variables to consider at each iteration increases, and decreases when the project is over-modularized. Myopically, as the number of decision variable to be considered in each iteration increases, the room of performance improvement increases, thus there is a higher possibility of attaining a greater peak performance. However, as the number of decision variables to be considered in each iteration increases, the number of local peaks increases, and it places more effort on the ISD team to discover the next performance peak. However, as previously discussed earlier, to fully realize the benefits that agile methodology brings, the ISD project needs to be of sufficient size. The adaptable and flexible benefits of the agile methodology are not realized by projects of very small size and scope. In contrast, when projects are of small size and scope, the predictability and uncertainty of the problem lessens. Thus, the predictable nature of the waterfall methodology suits such project cases better than the agile methodology.

We also find that the agile methodology is significantly more resilient to environmental factors – project uncertainty, complexity and resource constraints – as compared to the waterfall methodology. However, when environmental factors are strong – which causes the project landscape to be more rugged – the performance difference between the imperfect and non-modular structures shrinks. For projects of both imperfect and non-modular designs, the agile methodology is significantly more resilient to higher project uncertainty, complexity as well as higher resource constraints than the waterfall methodology. When environmental factors are strong, the project uncertainty is higher, as well as complexity. The predictable nature of the waterfall methodology works by considering all decision variables in each iteration, thereby requiring significantly more effort to achieve or maintain performance as compared to agile. In addition, as environmental factors become stronger, the predictability of the problems decreases significantly, defying the nature of the waterfall methodology. Consequently, as the environmental factors become stronger, the waterfall projects achieve a much lower performance yield. In contrast, the agile methodology simulates a "divide-and-conquer" strategy when developing ISD projects, by breaking down a large and complex problem into smaller chunks – in this case, modules – and completing the work on one module before moving on to the next (Elshamy and Elssamadisy 2006). Agile methodology also promotes adaptability, where changes can be made to modules that are completed to adapt improve performance. This strategy reduces the effort needed to maintain the project, facilitates changes and improves the overall project quality as compared to the waterfall methodology.

The implications discussed above can be summarized into a contingency matrix (see Figure 10). When a project team faces strong environmental factors (large scope, high uncertainty, high complexity), the resilience of agile makes it a more favorable ISDM to adopt, regardless of the modularity of the project design as the performance difference between imperfect and non-modular designs shrinks with strong environmental factors. However, when the team faces weak environmental factors (small scope, low uncertainty, low complexity), the project is rather predictable, and the waterfall methodology works well when the project structure is non-modular. In the case where the project structure is relatively modular with weak environmental factors, the performance gain from using agile is not significantly more than that of the use of the waterfall methodology. Thus in this case, both the agile and waterfall ISDMs are adequate.

Figure 10. ISDM Choice by Degree of Problem Modularity and Environmental Factors



In managerial terms, to attain the highest performance possible by using the agile methodology, it is crucial for ISD teams to ensure that they achieve an adequately modular and sized project at the initial starting point of planning. In reality, this accuracy will not be achieved in one project. The ISD team will have to go through several iterations of project planning and development to obtain the experience needed to determine how the project elements should be structured best for the use of agile. Planning is always imperative for project management. This planning phase should be structured such that the ISD team explore the various requirements of the ISD project and structure them accordingly in modules, and benchmark the results accordingly with past experiences on whether the planned modularity and size of the project is adequate for agile to be used.

This paper is not without limitations. Despite the advantages of simulation methods in their ability to incorporate complex dynamics without worrying about analytical tractability and their ability to study constructs of interest that may be difficult if not impossible to manipulate in field studies, simulation models are stylized theoretical models of reality that require rigorous validation through empirical testing. Also, the industry today adopts hybrid forms of both plan-based and agile ISDMs for various real-world ISD projects (Parsons and Lal 2006). However, we have only investigated *pure* forms of plan-based and agile ISDMs in this research. Extensions to this research involving hybrid forms of ISDMs will be expected to yield valuable insights. Nonetheless, the analysis of the pure forms of the ISDMs still provides us with a strong foundation to understanding the effects of modularity on the performance of ISDMs. That said, we believe that our *NK* landscapes model of ISDMs has allowed us to provide valuable theoretical insights that can guide further theoretical and empirical research.

Acknowledgements

This study was supported by the National University of Singapore Academic Research Fund (AcRF) Tier 1 Start Up Research Grant.

References

Amaral, L.A., and Uzzi, B. 2007. "Complex Systems: A New Paradigm for the Integrative Study of Management, Physical, and Technological Systems," *Journal of Management Science* (53:7), pp. 1033-1035.

Awad, M. A. 2005. "A Comparison between Agile and Traditional Software Development Methodologies," School of Computer Science and Software Engineering," The University of Western Australia, Australia.

Baldwin, C.Y., and Clark, K.B. 2000. *Design Rules: The Power of Modularity*, Cambridge, MA: The MIT Press.

Baldwin, C.Y., and Clark, K.B. 2001. "The Value and Costs of Modularity," Working Paper, Harvard Business School, Boston, MA.

Baldwin, C.Y., and Clark, K.B. 2004. "Modularity in the Design of Complex Engineering Systems," Working Paper, Harvard Business School, Boston, MA.

Boudreau, T., Tulach, J., and Wielenga, G. 2007. "The Benefits of Modular Programming," in *Rich Client Programming*, Upper Saddle River, NJ: Prentice Hall, pp. 9-19.

Brooks, F. 1975. *The Mythical Man-Month: Essays on Software Engineering*. Boston, MA: Addison-Wesley.

Burton, R.M., and Obel, B. 1995. "The Validity of Computational Models in Organization Science: From Model Realism to Purpose of the Model," *Computational and Mathematical Organization Theory* (1:1), pp. 57-71.

Brusoni, S., Marengo, L., Prencipe, A., and Valente, M. 2007. "The Value and Costs of Modularity: A Problem-Solving Perspective," *European Management Review* (4:2), pp. 121-132.

Cerveny, R.P., Garrity, E.J., and Sanders, G.L. 1990. "A Problem-Solving Perspective on Systems Development," *Journal of Management Information Systems* (6:4), pp. 103-122.

Cohen, K.J., and Cyert, R.M. 1965. "Simulation of Organizational Behavior," In *Handbook of Organizations*, J.G. March (ed.), Chicago, IL: Rand McNally, pp. 305-334.

Conley, C.A., and Sproull, L. 2009. "Easier Said than Done: An Empirical Investigation of Software Design and Quality in Open Source Software Development," in *Proceedings of the 42nd Hawaii International Conference on System Sciences*. January 6-9, Waikoloa Village, HI.

Davis, J., Bringham, C., and Eisenhardt, K. 2007. "Developing Theory Through Simulation Methods," *Academy of Management Review* (32:2), pp. 480-499.

Duffy, A., and Ferns, A.F. 1998. "An Analysis of Design Reuse Benefits," in *Proceedings of the 12th International Conference on Engineering Design*. August 24-26, Munich, Germany.

Duffy, A., Smith, J.S., and Duffy, S.M. 1998. "Design Reuse Research: A Computational Perspective," in *Proceedings of Engineering Design Conference on Design Reuse*. June 23-25, London, UK.

Dybå, T., and Dingsøyr, T. 2008. "Empirical Studies of Agile Software Development: A Systematic Review," *Information and Software Technology* (50:9-10), pp. 833-859.

Elshamy, A., and Elssamadisy, A. 2006. "Divide After You Conquer: An Agile Software Development Practice for Large Projects," in *Proceedings of the 7th International Conference on Extreme Programming and Agile Processes in Software Engineering*, June 17-22, Oulu, Finland, pp. 164-168.

Ethiraj, S.K., Levinthal, D., and Roy, R.R. 2008. "The Dual Role of Modularity: Innovation and Imitation," *Management Science* (54:5), pp. 939-955.

Geambasu, C.V., Jianu, I., Jianu, I., and Gavrila, A. 2011. "Influence Factors for the Choice of a Software Development Methodology," *Accounting and Management Information Systems* (10:4), pp. 479-494.

Hahn, J., and Lee, G. 2010. "The Effect of Knowledge Overlap, Task Interdependence, and Trust on IS Development Performance," in *Proceedings of the 2010 International Conference on Information Systems*, December 12-15, St. Louis, MO, Paper 199.

Hahn, J., Lee, G., and Howison, J. 2013. "A Theoretical Framework for Requirements Research: A Complex Adaptive Systems Approach using the *NK* Fitness Landscapes Model," Working Paper, National University of Singapore, Singapore, Singapore.

Kässi, T., Leisti, S., and Puheloinen, T. 2008. "Impact of Product Modular Design on Agile Manufacturing," *Mechanika* (74:6), pp. 56 -62.

Kauffman, S.A. 1993. *The Origins of Order: Self-Organizing and Selection in Evolution*. New York: Oxford University Press.

Kauffman, S.A., and Weinberger, E.D. 1989. "The *NK* Model of Rugged Fitness Landscapes and Its Application to Maturation of the Immune Response," *Journal of Theoretical Biology* (141:2), pp. 211-245.

Korpela, M., Mursu, A., and Soriyan, H. 2002. "Information Systems Development as an Activity," *Computer Supported Cooperative Work* (11:1-2), pp. 111-128.

Langlois, R.N. 2002. "Modularity in Technology and Organization," *Journal of Economic Behavior & Organization* (49:1), pp. 19-37.

Levinthal, D.A. 1997. "Adaptation on Rugged Landscapes," *Management Science* (43:7), pp. 934-950.

Mishra, D., and Mishra, A. 2011. "Complex Software Project Development: Agile Methods Adoption," *Journal of Software Maintenance and Evolution: Research and Practice* (23:8), pp. 549-564.

Modha, J., Gwinnett, A., and Bruce, M. 1990. "A review of information systems development methodology (ISDM) selection techniques," *OMEGA: the International Journal of Management Science* (18:5), pp. 473–490.

Narduzzo, A., and Rossi, A. 2003. "Modular Design and the Development of Complex Artifacts: Lessons from Free/Open Source Software," ROCK Working Papers, 21. Italy.

Nejad, G., and Irani, H. 2012. "Decentralized Principles: New Modular Software Development Principles, a Robust Object Oriented Approach," *International Journal of Computer Applications* (44:13), pp. 61-65.

Parsons, D., and Lal, R. 2006. "Hybrid Agile Development and Software Quality," in *Proceedings of the 14th International Software Quality Management Conference*, April 10-12, Southampton, UK, pp. 283-298.

Simon, H.A. 1996. *The Sciences of the Artificial*. Boston, MA: The MIT Press.

Standish Group 2009. "CHAOS Summary 2009," The Standish Group, Boston, MA.

Sweeney, T. 2003. IT Executives Take a More Modular Approach to CRM Software. May 30, Available at http://www.techrepublic.com/article/it-executives-take-a-more-modular-approach-to-crm-software, Retrieved September 1, 2014.

Ulrich, K., and Eppinger, S. 1999. *Product Design and Development* (2nd ed.). New York: McGraw-Hill.

Vavpotič, D., and Vasilecas, O. 2012. "Selecting a Methodology for Business Information Systems Development: Decision Model and Tool Support," *Computer Science and Information Systems* (9:1), pp. 135-164.

Xia, W., and Lee, G. 2005. "Complexity of Information Systems Development Projects: Conceptualization and Measurement Development," *Journal of Management Information Systems* (22:1), pp. 45-83.

Yan, X. T., Stewart, B., Wang, W., Tramsheck, R., Liggat, J., Duffy, A.H., and Whitfield, I. 2007. "Developing and Applying an Integrated Modular Design Methodology Within a SME," in *Proceedings of the International Conference on Engineering Design*, August 28-31, Paris, France.