An Integrative Economic Optimization Approach to Systems Development Risk Management

Michel Benaroch*
Syracuse University
Martin J. Whitman School of Management
721 University Avenue
Syracuse, NY 13224
Tel: (315) 443 3498
Fax (315) 443 5457
mbenaroc@syr.edu

and

James Goldstein
Syracuse University
Martin J. Whitman School of Management
721 University Avenue
Syracuse, NY 13224
jcgoldst@syr.edu

Acknowledgments. The authors thank the Brethen Institute for Operations Research at the Whitman School of Management, Syracuse University for partial funding.

Abstract – Despite significant research progress on the problem of managing systems development risk, we are yet to see this problem addressed from an economic optimization perspective. Doing so entails answering the question: what mitigations should be planned and deployed throughout the life of a systems development project in order to control risk and maximize project value? We introduce an integrative economic optimization approach to solving this problem. The approach is integrative since it bridges two complementary research streams: one takes a traditional micro-level technical view on the software development endeavor alone, another takes a macro-level business view on the entire lifecycle of a systems project. Bridging these views requires recognizing explicitly that value-based risk management decisions pertaining to one level impact and can be impacted by decisions pertaining to the other level. The economic optimization orientation follows from reliance on real options theory in modeling risk management decisions within a dynamic stochastic optimization setting. Real options theory is well suited to formalizing the impacts of risk as well as the asymmetric and contingent economic benefits of mitigations, in a way that enables their optimal balancing. We also illustrate how the approach is applied in practice to a small realistic example.

Keywords — D.2.9.m Risk management, K.6.1.f Systems development, K.6.0.a Economics.

* Corresponding author
1. Introduction

Despite significant research progress on the systems development risk management (SDRM) problem (Boehm, 1989; Fairley, 1994; Sullivan et al., 2001; Boehm, 2006a; Biffl et al., 2006; McManus, 2004; Lyytinen et al., 1998; Barki et al., 1993), we are still lacking an approach to addressing this problem from an economic optimization perspective. Such an approach, first and foremost, must allow to formally model two competing economic tendencies for various kinds of known risk mitigations, and facilitate the optimal balancing of the asymmetric impacts of these tendencies. Mitigations are deployed, and their cost is incurred, before it is known whether risk will occur and cause a problem. By contrast, the payoff from these mitigations’ ability to limit downside risk and/or enhance upside potential is contingent on the actual occurrence of risk and its associated problem. Also, the benefits of risk mitigations are non-additive because multiple mitigations could have a similar impact on the risk profile of a project. Without the ability to balance these two competing economic tendencies, it is impossible to know whether any set of mitigations designed for a target system development project is optimal or even adds value.

An adequate solution approach must also bridge two disconnected research streams, in light of a clear recognition by value-based risk management research that “risk analysis and risk management … pervade the entire information systems life cycle” (Boehm, 2006b, p. 114). One stream traditionally concentrates on micro-level technical aspects of executing a software development project with a focus on controlling cost and schedule risk (Boehm, 1989; Fairley, 1994). This stream, however, overlooks that software development is primarily a value-creation activity (Boehm and Sullivan, 2000, p. 15):

“… the objective of software design is to create surplus value. The goal is not to achieve verifiability, evolvability, safety, quality, usability, reusability, reliability, satisfaction of a formal specification, possession of a mathematical semantics, or any other technical property, per se. Technical properties are of course critical to creating value, but they are the means, not the ends.”

The other stream focuses on macro-level aspects concerning the role of project flexibility in controlling business and organizational risk throughout the entire life of a system, from the business case stage to the operations stage proceeding software development (Benaroch, 2002; Kumar, 2002). This stream, however, it is neither concerned with how flexibility comes about nor with how micro-level design and architectural choices can drive the amount of flexibility available. Clearly, these streams address complementary aspects of the same problem. In fact, value-based software engineering research argues that, to manage risk in a way that maximizes value creation, these research streams must be bridged relative to a host of activities ranging from business case analysis to software testing (Biffl et al., 2006; Boehm and Sullivan, 2000). Micro-level technical decisions to deploy certain mitigations (e.g., over-engineering, design for reuse) create flexibility that enables macro-level decisions to mitigate risk by investing partially
(pilot, incremental), disinvesting (contract, abandon) or reinvesting (expand, launch follow-up projects). By the same token, it is the anticipated need to make such macro-level business decisions that should be the primary driver of micro-level technical decisions. In sum, risk management needs arising at one level impact and can be impacted by risk management needs arising at the other level.

The goal of this paper is to introduce an integrative economic optimization approach to SDRM. The approach is integrative in the sense that it fits the micro-level technical view with the macro-level business view of SDRM. And its economic optimization orientation follows from its use of real options theory (ROT) to conceptualize and model decisions within a dynamic stochastic optimization setting.

Using ROT as the underlying modeling framework allows us to build on and contribute to a large body of related research on software design and architectural choices. Baldwin and Clark (1993; 2001) pioneered the use of ROT in systems design and engineering, showing that modular designs create flexibility which adds value in the form of real options. Sullivan et al. (1999), in turn, argued that ROT can provide insight about software modularity, phased project structures, delay of investment and software design decisions, and other software engineering strategies. Others similarly argued for the use of ROT in investigating the economics of COTS development (Erdogmus, 1999), investments in application frameworks (Favaro and Favaro, 1999), and software reuse infrastructure (Favaro et al., 1998). On the whole, though, this early work offers no explicit mathematical treatment of essential concepts, such as risk and mitigations, and how they relate to the cost of creating flexibility or the business value created by flexibility. Instead, it focuses on introducing the links between such essential concepts and ROT while illustrating the concepts using idealized examples. Nevertheless, this early work stimulated discussion of the broad relevance of ROT to value-based software engineering (Boehm and Sullivan, 2000, p. 4):

“… central concepts in software engineering, such as information hiding, architecture, the spiral model, and heuristics on the timing of software design decisions, have yet to be linked adequately to business value … such linkages can be … established in terms of the real options value of the decision flexibility afforded by modular designs and phased project structures … software engineers can develop a better understanding of … the value of new information that reduces uncertainty …; the present value of options whose payoffs depend on how exogenous uncertainties are resolved, including options to enter new markets … make follow-on investments … ship a product early …; the opportunity cost of investing early in the face of uncertainty …”

Significant progress has indeed been made in some of these directions. Most notable are the application of Baldwin and Clark’s theory to Parnas’ information-hiding technique to modularization around anticipated sources of change (Sullivan et al., 2001), the rationalization of agile development based on Extreme Programming practices (Erdogmus and Favaro, 2002), the valuation of staged software development under technical and market risk (Erdogmus, 2002), and the valuation and selection of stable middleware architectures (Bahsoon et al., 2005). These works are better
formalized mathematically, and some offer more realistic examples (Sullivan et al., 2001) or real-world case applications (Bahsoon et al., 2005).

Progress notwithstanding, important gaps remain. These gaps are discussed in detail later, and two key ones are highlighted here. First, while great attention is given to the risk management goals of anticipation and response to environmental change, other fundamental goals are unaddressed (Sullivan et al., 2001). One is manipulation of the root causes of environmental change. Another is proactive learning about change via mitigations (e.g., prototyping activities) that create new information and can impact the risk perceived by project managers and software designers; Erdogmus (2002), for example, looks at staged development that starts with a prototype, but accounts only for passive learning via deferral of decisions that allows for information to arrive with the passage of time. Second, the more formalized parts of extant work address only one mitigation at a time, such as information-hiding modularity (Sullivan et al., 2001) and middleware section (Bahsoon et al., 2005). Project managers, however, may use an entire arsenal of mitigations to tackle various risks at different project stages. What is crucial is that interactions among mitigations that are used across project stages, or even within one project stage, can influence their aggregate value, as Boehm (2006b, p. 123) explains in the case of Sullivan et al.’s (2001) work:

“… modularization around anticipated sources of change … can also be combined with other economic approaches, such as buying information to reduce the risk of anticipating the wrong set of changes (e.g., via prototypes, user surveys, marketplace watch, or technology watch activities).”

This paper tackles such gaps head-on in the course of developing an approach to addressing SDRM using multi-mitigation strategies from an economic optimization perspective. Importantly, while our exposition of the approach may appear aligned mostly with a project level perspective on SDRM, the approach is also of value at the level of addressing one type of mitigation or design principle from a more technical perspective. We devote less attention to the latter aspect, but will discuss how extant work, such as on middleware valuation and selection (Bahsoon et al., 2005), can be recast using our approach. This should clarify how our work both builds on and makes a contribution to extant research on value-based risk management. In particular, the contribution follows from the fact that many SDRM strategies are still based mostly on experience and heuristics not anchored in formal models (Sullivan et al., 1999). The paper also offers a small but realistic example of how our approach works in practice. Since the approach is new and was never presented elsewhere, the example is in keeping with the tradition of extant work to expose new ideas using simple examples that allow the reader to focus on essential concepts. Hence, empirical application and validation of our approach remains a crucial matter for future research.

The paper proceeds as follows. It describes the SDRM problem and fundamental risk management strategies,
and reviews the literature to identify key gaps in our ability to fully model these strategies. It next conceptualizes and formalizes notions underlying the SDRM problem in economic terms that fit real option concepts. It then discusses the option-based modeling of SDRM and presents an analytical model for finding the value of optional sequential mitigations aimed at controlling risk and maximizing value. It continues with an illustrative example involving several optional sequential mitigations, to show how the model finds the optimal subset of mitigations to plan and deploy for a given system development project. It concludes with a discussion of future research on estimating model parameters, scaling the model for any number of sequential mitigations, and developing tools for supporting our approach in real-world settings.

2. Theoretical Background

2.1. Risk Management as Economic Optimization

Analogous to financial risk management, SDRM is about favorably altering the probability distribution of the value of a systems development project. Figure 1 visualizes this idea in relation to two main activities: risk assessment identifies relevant risks and quantifies their impact on project value, and risk control identifies mitigations for controlling identified risks and plans when to apply them over the project lifetime. The aim of deployed mitigations is to lower the probability that any risk would occur and cause a problem and/or to lower the negative impact of the problem. Monitoring how risks evolve in light of deployed mitigations yields new information that may require re-iterating these activities.

![Figure 1](image)

Figure 1: risk management favorably alters the probability distribution of project value

Lower-level risk management goals are set in recognition that systems are developed to meet specific requirements in some operating environment. In this sense, the traditional goals of SDRM are to anticipate environmental changes that give rise to risk and design software in ways that offer the flexibility to respond to those
changes. Recent work recognizes the need to broaden the scope of SDRM to include manipulation of change as another goal (Sullivan et al., 2001, p. 9):

“accommodating environmental change is not limited to just anticipating change, …, but includes more generally both responsiveness to change and [proactive] manipulation of change.”

A fourth goal that is implicit in most discussions is proactive learning about environmental change. We explain these goals next and summarize their relationships in Figure 2.

**Figure 2**: generic risk management goals

*Anticipation* of change assumes that an existing risk might occur and cause a problem, and the aim is twofold. One is to deploy mitigations that build into the project flexibility that makes planned response mitigations (technically and economically) feasible. For example, mitigations such as over-engineering and information-hiding modularity offer flexibility to *expand* a project’s development scope or operational scope (Sullivan et al., 2001; Bahsoon et al., 2005), mitigations like task decomposition and scoping add flexibility to *contract* scope, and mitigations like design-for-reuse yield software assets with a higher acquisition-cost to resell-cost ratio and create flexibility to *abandon* a project in midstream (Favaro and Favaro, 1999). Another aim of anticipation is to lower the risk’s probability of occurrence or its potential impact. This can be accomplished by moving risk from one part of a project to another. For example, via such mitigations as task decomposition, staging a project allows to set milestones for reviewing information yielded on a completed part as well as to push a risky part up or down the timeline so as to reveal early if the entire project can be completed as planned or to gain time to take actions that lower risk (Boehm, 1989). *Incremental development* of small releases is a variant that permits deferral of changes to later releases and controls risks due to ongoing requirement changes, where each release also yields information on risks relating to performance shortfalls, customer and supplier adoption, etc. (Erdogmus and Favaro, 2002). Risk can also be transferred to a third party by *outsourcing* (via fee-based contracts) parts of a project to capable vendors as a way to shield against various development risks, or by *leasing* project resources (servers, networks, etc) with a possibility to break the lease in midstream and save the residual cost of resources in case risk materializes (Boehm, 1989).
Responsiveness to change assumes that an existing risk is likely to occur and cause a problem, and the goal is to respond to the problem in a way that limits its impact. This risk management goal could require the ability to contingently change the project’s initial goals, for example, by expanding or contracting the project scale (e.g., add or cancel features, use more or less development personnel) or its operational scope (e.g., enable or disable features) in response to positive or negative risks. In extreme cases, it could also involve contingency plans for recovering from risk, for example, by abandoning a project and redirecting its resources to alternate uses which are more valuable.

Manipulation of change assumes that an existing risk can be lowered or even eliminated by proactively manipulating its root causes. For example, if an implementation technology is risky because it lack maturity, manipulation could switch development to a tested product or a more mature technology. Other sample manipulation mitigations include staffing with top talent (Boehm, 1989) and lobbying so that a standard setting organization will deprecate interfaces rather than change them outright.

Learning about change aims at yielding information about the likelihood of risk problems and their associated impacts, since addressing all the previous risk management goals require “betting” on the occurrence of these problems. Learning could be passive via deferral of actions that allows for information to arrive with the passage of time (e.g., technology reviews), or proactive via deployment of mitigations that generate new information (e.g., user surveys, change management programs). Proactive learning also occurs when executing parts of a project, via prototyping activities (e.g., performance tests, feasibility studies, technology compatibility studies) that yield information on how technical risks can influence project success, or by piloting a scaled-down operational project that also yields information about how external and internal parties (e.g., end-users, suppliers, regulatory bodies) might react to the full-scale project.

2.2. Economics of Flexibility and Real Options
We next examine the degree to which existing work on real options theory (ROT) can address the four SDRM goals in Figure 2. We highlight notable gaps in this work that are the target of our present paper. Part of the work reviewed holds a business view on SDRM while another part holds a technical view.

2.2.1. “Business” View on Flexibility, Options and IT Project Investments
Work related to the business or macro view of SDRM argues that flexibility to change the course of a project can be valued as real options (Trigeorgis, 1996). A real option is a right, not obligation, that a manager implicitly holds (e.g., defer project) or deliberately makes possible by expensing some cost (e.g., over-engineer to enable expansion).
When deciding, for example, on project launch timing (Benaroch and Kauffman, 1999) or if to accept a project (Taudes et al., 2000), it is crucial to value the flexibility embedded in a project and to incorporate this value into the decision. Various studies have applied this thinking to different real options, including: defer, pilot, prototype, stage, change-scale, lease, and growth (Benaroch et al., 2006). Some of this work also looks at option-based risk management (Benaroch, 2002; Kumar, 2002), with a focus on the question: Where, how much and what forms of flexibility to build into a project so as to control risk and maximize value?

This business-related work has valuable strengths and notable weaknesses. One strength is the identification of different forms of flexibility with different real options, which can impact the distribution of project value in more than one way (Benaroch 2002). Another is the recognition that systems projects usually embed interacting, compound options whose value is not additive and treating them as such (Benaroch et al., 2007). A third strength is empirical support for the logic of option-based risk management, demonstrated in a study of project escalation behavior (Tiwana et al., 2006) and a study of the forms of flexibility (real options) and respective mitigations built into actual risk management plans for large systems projects (Benaroch et al., 2006). A major weakness of this work, however, is that it is not concerned with how flexibility comes about in systems projects. In particular, it is not concerned with how the amount of flexibility present in a systems project is impacted by software engineering decisions, such as, how much to invest in modularity.

2.2.2. “Technical” View on Flexibility, Options and Software Development

Work related to the technical or micro view of SDRM argues that ROT is ideal for approaching many software design issues from an economic perspective. As we said earlier, part of this work has focused on introducing and illustrating essential concepts using idealized examples, but another part has made much progress and deserves a closer look.

One work worthy of discussion is Sullivan et al.’s (1999) and Boehm and Sullivan’s (2000) real options interpretation of the spiral development model. On the premise that this model imposes a phased project structure aimed at anticipating and responding to environmental change (risk), these authors use two central real option concepts to frame this model as a problem concerned with value-maximization under uncertainty. One concept is that phased projects create series of compound options, as undertaking one phase yields new information about risk and creates an option (or flexibility) to invest or not in the next phase. Another concept is that it is logical for early phases to invest in prototypes, user surveys and other proactive learning mitigations that yield information for resolving risk and developing alternative ways to proceed. But, the authors do not formalize these concepts or the
overall value-maximization problem, and instead offer just a partial and simplified example with arbitrary parameter values. As such, no attention is given to the cost of creating flexibility or to the non-additive nature of compound options. Most notably, no effort is made to model the value of proactive learning. In fact, the authors argue that ROT must be supplemented by decision theoretic tools for analyzing the value of information associated with proactive learning. We will show, though, how this value can be modeled using ROT.

Another work is Sullivan et al.’s (2001) application of Baldwin and Clark’s (2001) theory to information-hiding modularity. It equates the flexibility offered by a modular design with a reduction in the number of interacting software modules that must be reworked in response to anticipated environmental change, and uses Baldwin and Clark’s (2001) real options model to value the flexibility offered by different modularizations. The authors also use Parnas’ benchmark KWIC problem as an example, but acknowledge that their model’s results are “back-of-the-envelope predictions” (p. 4) and its parameters rest on “back-of-the-envelope assumptions” (p. 7). More importantly, this work offers no way of linking flexibility to the creation of monetary or other forms of business value; instead, it measures value in terms of how much more flexibility does one modular design offer relative to some base strawman design. This is so, in part, because the authors do not measure or even define a central notion – risk – in explicit terms relating to (business) value. Furthermore, like Baldwin and Clark, they ignore the cost of designing modularizations but explicitly recognize the crucial need to account for it. Last, the authors explicitly liken the flexibility of a multi-component modularization to a set of compound options, but acknowledge the inadequate treatment of these options as additive.

A third relevant work is Erdogmus and Favaro’s (2002) examination of the value of flexibility inherent in extreme programming (XP), a lightweight development process positioned to respond to change. These authors use simplified examples to show that the most widely publicized practices of XP create real options, which allow viewing XP as a value maximization process. One is the ‘Earn Early, Spend Late’ practice, according to which small initial investments and frequent releases create more value than large investments and mega-releases. The authors rationalize this practice by pointing out that, like in a phased project, each release provides for learning and creates an option to develop or not develop the next release, and releases can be ordered to implement higher-value features early so as to maximize the realized value. Another practice is ‘You Are not Going to Need It,’ according to which delaying development of a fuzzy feature creates more value than developing the feature early. The authors rationalize this practice by the value of deferral options it implies. Based on such examples, the authors argue that XP development shifts the emphasis from just reducing risk by anticipating change to also managing risk by
creating flexibility to respond to change. Overall, though, this work offers no mathematical treatment of its arguments and instead anchors them in heuristics that lack any formal grounding. As such, it is subject to the same weaknesses we have seen so far, namely: overlooking the cost of flexibility and ways to link flexibility with the creation of business value, no formal modeling of the value of proactive learning, and the treatment of compound options as additive.

The last work worth mentioning is Bahsoon et al.’s (2005) use of real option concepts to inform the selection of the most stable middleware architecture from among a fixed number of middleware candidates. These authors analogize the flexibility of a middleware candidate to meet an anticipated non-functional requirement change to an ‘architectural potential’ that is acquired contingent on paying the cost of accommodating that change on that middleware. Given a known set of anticipated changes, they use the standard Black-Scholes option model (Trigeorgis, 1996) to value the architectural potential of every middleware candidate relative to every anticipated change, and the candidate that maximizes the yield of its embedded flexibility is chosen. A key strength of this work is a recognition that multiple perspectives can associate different benefits with architectural potential, but that these benefits are eventually “cast into monetary value” (Bahsoon, 2005, p. 131), whether they are technical benefits (e.g., savings in development effort, improvements in personnel productivity) or business benefits (e.g., reduction in expansion, cost or in time-to-market). And, these benefits are modeled using a distribution (e.g., optimistic, likely, and pessimistic). However, considering that anticipated requirement changes can interact, in which case the value of options associated with accommodating these changes is non-additive, this work gives no explicit reason for treating these options as having additive values.

2.3. What is Missing?

Clearly, much work has used ROT as a coherent framework for conceptualizing and modeling various aspects of the SDRM problem, but open issues hinder the ability to address the SDRM problem from an economic optimization perspective. These issues are at the heart of the present paper.

The first issue has to do with the recognition that project managers have an entire arsenal of mitigations at their disposal, where they may apply some of these sequentially or in parallel in order to handle different risks at different project stages. Yet, existing work that is more mature addresses just one type of mitigation at a time, such as information-hiding modularity (Sullivan et al., 2001) and middleware section (Bahsoon et al., 2005). Unfortunately, work concerned with multi-mitigation strategies, such as the spiral model (Sullivan et al., 1999) and
agile XP development (Erdogmus and Favaro, 2002), is not formalized to the point required to address SDRM from an economic optimization perspective.

A related issue is how to model within the same economic framework the value of mitigations associated with all four generic SDRM goals (Figure 2). For the goal of manipulating change, no attempt has been made to use ROT to value the flexibility to deploy mitigations aimed at manipulating the root causes of change and risk. Likewise, for the learning goal, it is not yet clear how to value learning that is contingent on the proactive deployment of mitigations aimed at yielding new information for resolving risk (e.g., prototyping activities, user surveys). Existing work only values passive learning due to deferral of decisions that allows for information to arrive with the passage of time (Erdogmus and Favaro, 2002). Even for the anticipation and response goals, extant work does not account for the reciprocal relationship between the value of flexibility-creating anticipatory mitigations (e.g., over-engineering) and the value of flexibility-exploiting response mitigations (e.g., expand project scope).

The third open issue is the apparent need to account for the cost associated with creating flexibility and deploying mitigations that it enables. None of the existing work addresses this issue, except for Bahsoon et al. (2005) who employ a standard real option valuation model of the kind we build upon in this paper.

The last open issue is the need to handle the deployment of multiple mitigations that may have synergetic or substitutive impacts on value creation. This requires modeling the associated options as interacting options having a non-additive value. As we have seen, even situations involving a single type of mitigation can require valuing flexibility using compound options, as when selecting a middleware candidate for handling multiple anticipated changes (Bahsoon et al., 2005). In fact, many aptly recognize that options-oriented formulations of software design issues often involve a set of embedded compound options (Sullivan et al., 1999; Sullivan et al., 2001; Erdogmus and Favaro, 2002).

The following sections develop and illustrate an option-based conceptualization and formalization of the SDRM problem that addresses these issues.

3. Conceptualizing the SDRM Problem

This section lays down the conceptual elements of SDRM. These elements are later formalized within the real options modeling framework.

3.1. Problem Setting

Let $\tilde{V} = \tilde{P} - \tilde{I}$ be the stochastic (net) project value, where $\tilde{P}$ and $\tilde{I}$ are the project payoff and cost, respectively.
\( \tilde{P} \) and \( \tilde{I} \) are stochastic due to uncertain project attributes such as the product complexity, volatility of implementation platform, and capability of assigned programmers. To clarify, consider the ‘programmer capability’ attribute. Uncertainty over this attribute could make the project cost \( I \) uncertain and raise concerns about the quality of the project’s end product. This, in turn, could create uncertainty over the product usage level by end-users, and eventually impact the project payoff \( P \).

Every project attribute is numeric or can be mapped to a numeric scale. This is consistent with much software development research, especially on cost estimation models. For example, consider the COCOMO-II model (Boehm et al., 2000). It estimates project cost using 22 project attributes, all rated on the qualitative scale \{very-low, low, nominal, high, very-high, extra-high\} but this scale is mapped to a different range of numeric values for each attribute. For instance, two of the values of the Analyst Capability attribute are ‘very-low’=1.42 and ‘very-high’=0.71, and respective values for the Product Complexity attribute are ‘very-low’=0.70 and ‘very-high’=1.30. Where the nominal rating is applicable to an “average” software development project, COCOMO uses the specific ratings of each attribute to adjust up or down the cost of a target project relative to the cost of a comparable nominal project. Of course, project managers rarely are able to rate every project attribute with certainty.

In line with finance and decision sciences research (Elton and Gruber, 1995), an uncertain project attribute, denoted \( \tilde{A}_j \), is associated with a risk factor. A risk factor, denoted \( \tilde{r}_j \), is a stochastic variable defined over the range of possible deviations of the actual levels of \( A_j \) from its expected level:

\[
\tilde{r}_j = \{ y | A_j^{\text{actual}} - A_j^{\text{expected}} \}
\]

Each deviation of \( A_j \) from its expected level represents a negative (or positive) risk event that causes \( \tilde{V} \) to deviate from its expected value. For example, if the actual Programmer Capability of assigned personnel is ‘low’ (or ‘high’) instead of the expected level of ‘medium’, project cost \( I \) would end up higher (or lower) than expected. The risk associated with \( \tilde{r}_j \) is the variance of this risk factor, denoted \( \sigma_{r_j} \).

---

1 The simplified COCOMO-II model for estimating effort in person-months, is:

\[
E = A \times \text{size}^{1.01+0.01\sum_i SF_i} \times \left( \prod_{j=1}^{17} EM_j \right)
\]

where size is the project size, \( A \) is a constant, and \( EM_j \) and \( SF_i \) are project attributes. \( SF_i \)’s are attributes called scale factors because they account for the relative (dis)economies of scale for different size projects. \( EM_j \)’s are attributes called effort multiplier because they inflate or deflate the estimated cost (i.e., values of \( EM_j \) are 1 for nominal ratings, and lower [higher] than 1 if their impact is negative [positive] on project effort).
The risk of project value is due to exposure of the project payoff $\tilde{P}$ and cost $\tilde{I}$ to multiple risk factors, $\tilde{r}_1, \ldots, \tilde{r}_n$. If the subsets of factors influencing $\tilde{P}$ and $\tilde{I}$ are denoted $J_P$ and $J_I$, respectively, and assuming that these factors are independent, the risk of $\tilde{P}$ and $\tilde{I}$ is $\sigma_P = \sum_{j \in J_P} \sigma_{r_j}$ and $\sigma_I = \sum_{j \in J_I} \sigma_{r_j}$, respectively, where $\sigma_{r_j}$ is the risk of factor $\tilde{r}_j$. The independence assumption is consistent with much software development research (e.g., Boehm, 1989; Barki et al., 1993). As $\tilde{P}$ and $\tilde{I}$ could be exposed to one or more common risk factors, they can have a correlation, denoted $\rho_{PI}$. Thus, the risk of a project can be equated with the variance of $\tilde{V}$, denoted $\sigma_V$, which is a function of $\sigma_P$, $\sigma_I$ and $\rho_{PI}$.

Figure 3 depicts the main elements in our problem conceptualization.

![Figure 3: modeling of problem elements](image)

### 3.2. Risk Mitigations and their Impacts

A risk mitigation, denoted $M_i$, is an action aimed at increasing project value $V$ by favorably altering the distributions of $\tilde{P}$ and $\tilde{I}$. Mitigations associated with the four generic SDRM goals – manipulation, learning, response, and anticipation – have different impacts on the problem elements defined so far.

#### 3.2.1. Manipulation Mitigations

A manipulation mitigation impacts the root causes of a risk factor $\tilde{r}_j$ by changing the range of values of the underlying uncertain attribute $\tilde{A}_j$. For example, consider a case having uncertainty over the Product Complexity associated with a project. Let this attribute be defined as $A_j \in \{\text{very-low}=0.7, \text{low}=0.85, \text{nominal}=1.0, \text{high}=1.15, \text{very-high}=1.3, \text{extra-high}=1.65\}$; these are the numeric values COCOMO uses. If $A_j^{\text{expected}} = \text{‘high’}$, matching the
expected level of product complexity, but the assumption is that $A_{ij}^{\text{actual}}$ could fall between ‘low’ and ‘extra-high’, then the risk event $A_{ij} \neq \text{‘high’}$ implies that:

$$\tilde{r}_j \in \{A_{ij}^{\text{possible}} \setminus \text{‘high’}\} = \{-0.3, -0.15, 0.0, 0.15, 0.5\}$$

$$\mu_{\tilde{r}_j} = 0.04 \quad \sigma_{\tilde{r}_j} = 0.31$$

Consider a mitigation $M_i = \text{‘scoping’}$ that can treat the root cause of uncertainty by excluding a problematic part of the project. This lowers the expected product complexity from ‘high’ to ‘nominal,’ limits the possible complexity to the ‘very-low’–‘very-high’ range, and the range of values for $r_j$ becomes:

$$\tilde{r}_j \in \{A_{ij}^{\text{possible}} \setminus \text{‘nominal’}\} = \{-0.3, -0.15, 0.0, 0.15, 0.3\}$$

$$\mu_{\tilde{r}_j} = 0.0 \quad \sigma_{\tilde{r}_j} = 0.24$$

As seen in Figure 4, $M_i$ changes the number of negative risk events and their probabilities. To complete the example, assume for simplicity that $M_i$ is costless and that the payoff $P$ is deterministic but cost $\tilde{T}$ is stochastic and positively proportional to $r_j$. Then, the altered distribution of $\tilde{V} = P - \tilde{T}$ has a lower probability that $V$ will be below a threshold $\theta$ under which the project becomes unattractive (Figure 4).

### Figure 4: impact of manipulation mitigations
(choice of distributions is arbitrary and used only for illustration)

#### 3.2.2. Learning Mitigations

A learning mitigation yields new information about the probability and consequences of risk occurrence. It can involve passive learning while waiting to make an investment or active learning via a partial investment (e.g., prototype, market survey). For example, say a project is expected to generate payoff $P$ with the distribution seen in Figure 5a. According to the top branch of the decision tree in Figure 5c, the expected payoff is $210$ and its risk (standard deviation) is $104.4$. Suppose that a learning mitigation may help us to better assess the expected payoff. Historical data about the effectiveness of this mitigation appears in Figure 5b as a reliability matrix, which shows the percentage of cases where the mitigation predicted that state of nature $S$ is going to happen and $S$ actually happened.
The matrix lists conditional probabilities $p(S_M|S)$, for instance: probabilities in the first column are $p(\text{High}_M|\text{High})=0.8$, $p(\text{Med}_M|\text{High})=0.1$, $p(\text{Low}_M|\text{High})=0.1$. Using this matrix and Bayes Law, we can derive the reverse probabilities, $p(S|S_M)$, or the chance that the mitigation would predict $S$ is going to happen and $S$ will actually happen (Daellenbach, 1994). The result is seen in the lower branches of the decision tree in Figure 5c: the mitigation does not change the mean of $P$ but it lowers its standard deviation to 84.8. Overall, the impact of the mitigation is one of shrinking the bad tail of $P$’s probability distribution, as seen in Figure 5d.

![Figure 5](image)

**Figure 5**: impact of learning mitigations

(Choice of distributions is arbitrary and used only for illustration)

### 3.2.3. Anticipation and Response Mitigations

Anticipation and response mitigations are discussed together because of their reciprocal relationship. Anticipatory mitigations can have two aims: (1) to lower the consequences of a problem caused by risk occurrence, and/or (2) to lower the probability that the risk will occur.

Relative to the first aim, one type of anticipatory mitigations build into the project flexibility for ensuring the feasibility of deploying response mitigations that can lower/leverage the negative/positive impacts of a risk problem. Thus, unlike manipulation mitigations, the value of anticipatory mitigations depends on planned but contingent response mitigations. As we saw earlier, pairs of such mitigations could include: ‘over-engineering’ and an ability to ‘expand’ a system or ‘launch a follow-up project’ (in response to an opportunity), and ‘task decomposition’ and an ability to ‘contract’ development (in response to a problem). The response mitigations in
these pairs (e.g., expand, contract) alter favorably the distribution of project value by either shrinking its “bad” tail or enhancing its “good” tail (see Figure 6).

For example, consider a Web-based project with random payoff \( \tilde{P} \) (where \( \tilde{V} = \tilde{P} - I \)) that is influenced by risk factor \( r_j \) = ‘customer demand’. Demand in excess of expectation is a positive risk for which one suitable response could be to add new web site features for growing the project payoff \( P \). With a base design using HTML scripts, it would be possible to expand the site by time \( \tau \) at cost \( I_e \) to increase \( P \) by \( e \% \). The value of this contingent expansion is: \( \max(0, eP - I_e; \tau) \). Now, deploying a mitigation \( M \) that uses Java instead of HTML scripts would yield a more modular software design. In turn, this would enable site expansion to be completed at a lower cost \( I_e' \) (\( I_e' < I_e \)) and earlier time \( \tau' \) (\( \tau' \leq \tau \)), while increasing \( P \) by a larger percentage \( e' \% \) (\( e' > e \)). Thus, \( \max(0, e'P - I_e'; \tau') > \max(0, eP - I_e; \tau) \), and deployment of mitigation \( M \) (ignoring its cost) could be said to favorably alter the distribution of \( V \) by pushing its right tail upwards (Figure 6).

![Figure 6: impacts of anticipation and response mitigations](image)

(Choice of distributions is arbitrary and used only for illustration)

Relative to the second aim, lowering the probability of a risk problem, anticipatory mitigations of another type transfer risk from one part of the project to another or to a third party (e.g., vendor). This impact can be illustrated for the case of a ‘task decomposition’ mitigation, which provides for a staged development. Assume a project that yields payoff \( P \) and can be developed in two stages costing \( I_1 \) and \( I_2 \), respectively. Without task decomposition, staging is not possible and the project value can be written as: \( V = P - (I_1 + I_2) \). But, if task decomposition allows dividing the project into two stages, where stage 1 includes a risky part on which the entire project success hinges, a gateway review of information yielded by stage 1 about the risky part (e.g., about performance shortfalls) will determine whether to continue to stage 2 or to kill the project in midstream and save cost \( I_2 \). Respectively, the value of the staged project is: \( V = \max(P - I_2, 0) - I_1 \); here, \( I_2 \) is contingent on the outcome of stage 1. Hence, mitigations like ‘task decomposition’ facilitate a favorable altering of \( V \)’s distribution by pushing the “bad” tail downwards (Figure 6). The same logic applies for transferring risk to a third party, via development
outsourcing or resource leasing, where the equivalent of $I_2$ is the residual cost of completing the outsourced development effort or the lease payments.

To summarize the conceptualization of the four types of mitigations, the impact of all mitigations is one that favorably alters the distribution of project value $V$, by reducing the risk (variance) of $P$ and/or $I$ or by pushing the mean of $P$ right and/or the mean of $I$ left. Before we continue, it is important to note that deployment of any mitigation intended to combat a specific risk factor may aggravate other risks or even create new risks. For instance, consider a mitigation that switches development from HTML scripts to Java, to yield a more modular software design and increase expansion flexibility. All else equal, this mitigation may create a new secondary risk with respect to relative productivity of development programmers and maintainers with Java versus HTML scripts, and to the resulting cost differential. It is in such cases that the risk management process depicted in Figure 1 indicates a need to re-iterate the process with the deployment of mitigations and the arrival of new information about risk.

### 3.3. Basic Problem Formulation and Challenges

So mitigations can change the distribution, or distributional parameters, of one or more stochastic state variables, whose impacts on the project value $V$ we seek to manage. Figure 7 visualizes the way that sequential deployment of several mitigations can impact distributional parameters of $V$. The impact on the variance of $V$ reflects an anticipated reduction in project risk, or variability of $V$, conforming to the typical intuition that risk management is associated with a cone of uncertainty that gets progressively narrower as a project moves along the time line (Browning et al., 2002). However, Figure 7 also reflects the possibility that some mitigations can enhance (broaden) the good tail of the distribution of $V$. Thus, another impact of sequential mitigations on the mean of $V$ could be a progressive increase in the expected project value. Taken together, the impacts on the variance and mean of $V$ give rise to the possibility that mitigations do not necessarily shrink the distribution but just favorably alter it.

![Figure 7: impact of sequential mitigations on the expected value and risk (variance) of V](image-url)
As some mitigations may be deployed before a project has been approved (e.g., buy information to facilitate a wiser accept-reject decision), a project will hereafter be referred to as a project opportunity, denoted \( F(V(\tilde{P}, \tilde{I})) \), that can be completed on or before time \( T \).

Keeping in mind the way in which mitigations may impact the distribution of \( V \), by impacting the distributions of \( P \) and/or \( I \), the SDRM problem is about finding an optimal set of mitigations and their deployment timing that maximize the project opportunity value. This problem can be written as:

\[
\max_{i,t(i)} F(V(\tilde{P}, \tilde{I})| M_i, t(i), c_i) \tag{1}
\]

where

1. There are multiple optional mitigations.
2. Optional mitigation \( M_i \) can be activated for cost \( c_i \) at a pre-specified time \( t(i) \), \( 0 \leq t(i) < T \).

There are four challenges in formalizing this problem. First, the benefit of a mitigation \( M_i \) is asymmetric with the cost of \( M_i \): the cost is incurred before risk causes a problem, but the benefit is contingent on the actual occurrence of a problem. Second, the benefit of \( M_i \) may be sensitive to the deployment timing of \( M_i \). Third, the benefits of mitigations are non-additive, as the joint impact of multiple mitigations on distributional parameters of project cost \( I \) or payoff \( P \) may not be additive. Last, the above challenges are more severe when \( P \) and \( I \) are correlated due to exposure to common risk factors.

4. Real Options Formalization
This section explains how ROT enables modeling problem (1) in light of its challenges. It reviews the basics of analytical option models, formalizes the impact of mitigations on the distributional parameters of project value, and explains how the option models are adapted to handle up to two mitigations. It also reveals why more mitigations require numeric solution models on which we elaborate in a later section.

4.1. Option Models with no Mitigations
The main ideas underlying the real options thinking are as follows. A real option is a right, not obligation, to take an action depending on how uncertainty unfolds. The value of a project opportunity embedding a real option has an asymmetric probability distribution, as the option affords the flexibility to change the project course or to deploy mitigations in response to risk. A project can embed several real options.

Option models equate a project opportunity with flexibility to acquire stochastic payoff \( \tilde{P} \) upon spending stochastic cost \( \tilde{I} \) by time \( T \) at which the project can be completed (Trigeorgis, 1996). With no mitigations, only
passive learning is possible based on information arriving with the passage of time. This is the basic case of a project opportunity embedding only the flexibility, or option, to defer investment until $T$. The value of the opportunity, denoted $F(\tilde{P}, \tilde{I})$, is equated with that of a call option, written as:

$$F(\tilde{P}, \tilde{I}) = \text{present value} \left( E[\max(\tilde{P} - \tilde{I}, 0)] \right)$$

To find $F(\tilde{P}, \tilde{I})$, common option models make two key assumptions. One is that of a risk-neutral world, in which decision-makers’ risk preferences are irrelevant since they are indifferent between a risky project and a safe project having the same expected rate of return. This assumption implies that the value of flexibility is discounted at the risk-free interest rate, denoted $r$. The other assumption is that $\tilde{P}$ and $\tilde{I}$ are log-normally distributed state variables following the next Geometric Brownian Motion processes:

$$\frac{dP}{P} = (\mu_P)dt + \sigma_Pdz \quad \text{and} \quad \frac{dI}{I} = (\mu_I)dt + \sigma_Idz$$

where $\mu_x$ is the drift (growth) rate of state variable $x \in \{P, I\}$; $\sigma_x$ is the standard deviation, or volatility, of $x$; and $dz$ is an increment to a standard Wiener process describing the exogenous uncertainty of $x$. The log-normality assumption is relaxed in other models we discuss later (Datar and Mathews, 2004), but it fits our context well. That $P$ is log-normal is reasonable as the project payoff can drop only to zero but it can grow very large (Benaroch and Kauffman, 1999), and $I$ has been shown to be log-normal relative to software projects’ cost (DeMarco, 1982) and duration (Little, 2004). With this said, researchers and practitioners are known to also use Rayleigh and Weibull density functions (Kan, 2003). We will hence discuss later an alternative model that permits using any kind of probability distributions for $P$ and $I$.

The most basic option model for valuing a project opportunity as a call option is the Black-Scholes model (see Appendix A.1 for details). It assumes a stochastic asset (payoff) $\tilde{P}$ with volatility coefficient $\sigma_P$ and a deterministic exercise price (cost) $I$. We write this model’s option value as:

$$F(\tilde{P}, I) = P \cdot p_1 - Ie^{-rT} \cdot p_2$$

Here, if the project opportunity is worth undertaking, its terminal value is $P_T - I$ and its present value is $P - Ie^{-rT}$, where

---

2 A European call (put) on asset $P$ gives its holder a right to buy (sell) $P$ for an agreed exercise price $I$ at a fixed date $T$. For instance, a ‘June 08 call’ on IBM stock with a $75 exercise price allows to buy IBM shares for $75 on June 15, 2008; it is worth exercising only if an IBM share exceeds $75 on June 15. An American option is one that can be exercised at any time $t$, $t \leq T$.  

$e^{-rT}$ is the continuous time discount factor. To consider the possibility that the project can be unattractive to undertake ($P_T - I < 0$), $P$ and $I$ are weighted by the respective probabilities $p_1$ and $p_2$, as defined in Appendix A.1.1. Two noteworthy properties of this model are: the option value $F$ grows with the drift rate $\mu_p$ and with the variance $\sigma_p$ of $P$, $\partial F/\partial \mu_p > 0$ and $\partial F/\partial \sigma_p > 0$, respectively.

A variant of this model, called the asset-for-asset exchange model (see Appendix A.1.2), assumes that $\tilde{I}$, too, is stochastic with volatility coefficient $\sigma_i$, and that the correlation between $\tilde{P}$ and $\tilde{I}$ is $\rho_{PI}$. It computes the value of an option to exchange $\tilde{P}$ for $\tilde{I}$ as a call option on $\tilde{P}$ with exercise price $I$. We write the option value as:

$$F(\tilde{P}, \tilde{I}) = Pe^{-rT} \cdot p_1 - 1e^{-rT} \cdot p_2$$

where probabilities $p_1$ and $p_2$ are also a function of $\rho_{PI}$, as defined in Appendix A.1.2.

### 4.2. Modeling the Impact of Mitigations

To see what is needed to handle mitigations, recall that mitigations favorably alter the distributions of $P$ and/or $I$. A mitigation $M_i$ aimed at controlling the behavior of stochastic state variable $x$ can be said to lower $x$’s risk or volatility, $\sigma_i$, and/or increase $x$’s drift rate, $\mu_i$. But, doesn’t this impact of $M_i$ contradict the fact that options models compute an option value $F(\cdot)$ for which $\partial F/\partial \mu_x > 0$ and $\partial F/\partial \sigma_x > 0$? In other words, while $F(\cdot)$ is such that it increases with risk ($\sigma$), if a mitigation $M_i$ lowers risk, then $M_i$ will actually lower the value $F(\cdot)$. Sullivan et al. (2001) offer a more striking description of this paradox:

“because the value of an option increases with technical potential (risk), modularity creates seemingly paradoxical incentives to seek risks in software design, provided they can be managed by active creation and exploitation of options.”

In this light, it is crucial to see how we reconcile this paradoxical phenomenon in a way that is consistent with the logic of real options and risk management.

Following Koussis et al. (2004), let the impact of mitigation $M_i$ be denoted $k_i$, and let $k_i$ itself be lognormal with mean $\gamma_i$ and volatility $\sigma_i$. Respectively, the stochastic process in (2) is adapted into:

$$\frac{dx}{x} = \mu_x dt + \sigma_x dz + \sum_{i=1}^{N}(k_idq_i).$$

Where $x \in \{\tilde{P}, \tilde{I}\}$ is a state variable, parameters $\mu_x$, $r$, $\sigma_x$ and $dz$ are as in (2), $k_i$ is the (time-independent) lognormal impact of $M_i$ on $x$, $dq_i$ is a control variable equaling 1 if $M_i$ is deployed and 0 otherwise, and $N$ is the number of mitigations. This process assumes that a mitigation $M_i$ impacts state variable $x$ proportionately: it
multiplies $x$ by $1+k_i$, where the impact parameters $\gamma_i$ and $\sigma_i$ are quantities relative to $\mu_x$ and $\sigma_x$, respectively. As seen in Figure 8, conceptually, the distributions of $x$ before and after applying $M_i$ are $\ln(x) \sim N(\mu_x, \sigma_x)$ and $\ln(x|M_i) \sim N(\mu_x + \gamma_i, \sigma_x + \sigma_i)$, respectively, and the value of a project opportunity with $M_i$ deployed would be greater, $F(x|M_i) > F(x)$. Hence, by adding $\sigma_i$ to the volatility of $P$, $\sigma_P$, and/or of $I$, $\sigma_I$ the value of an anticipated risk reduction in $\sigma_P$ and/or $\sigma_I$ will be reflected in the increase in the project opportunity value, and the value of anticipated risk reduction is the difference $F(x|M_i) - F(x)$. This simple observation explains how we can reconcile the seemingly paradoxical nature of real option models.

![Figure 8: modeling the impact of a mitigation](image)

Importantly, where $M_i(\gamma_i, \sigma_i)$ denotes the mean and volatility impact parameters of mitigation $M_i$, parameters $\gamma_i$ and $\sigma_i$ can be set differently for different types of mitigations. The parameter values would depend on whether the target mitigation is intended to impact only $P$, only $I$, or both $P$ and $I$. In particular, all mitigations that impact $P$ and/or $I$ would have a volatility impact of $\sigma_i^2 \geq 0$, but mitigations that impact only $P$ or only $I$ would have a mean impact of $\gamma_i \geq 0$ or $\gamma_i \leq 0$, respectively.

### 4.3. Option Models with Mitigations

We can now see how the basic option models are adapted to handle mitigations. We start with Koussis et al.’s (2004) adaptation for a single mitigation that impacts project payoff $P$, extend it for two parallel mitigations, one impacting $P$ and one $I$, and conclude with an adaptation for two sequential mitigations, both impacting $P$. As we will see later, handling three or more mitigations requires adaptation of numeric option models.

**One Mitigation.** Assume a project opportunity with a stochastic payoff $\widetilde{P}$, a deterministic cost $I$, and a mitigation $M_0$ that costs $c_0$ and impacts $\widetilde{P}$ with impact parameters $(\gamma_0, \sigma_0)$. The value of the opportunity conditional
on deployment of $M_0$ at $t=0$ is denoted (see Appendix A.2.1 for details):

$$F(P, I | M_0) = Pe^{\gamma_0} \cdot p_1 - I e^{-rT} \cdot p_2 - c_0$$  \hspace{1cm} (6)$$

Compared to model (3) where no mitigation is deployed, the impact of $M_0$ on the distribution of $P$ makes probabilities $p_1$ and $p_2$ also a function of $(\gamma_0, \sigma_0)$, boosting $p_1$ up and $p_2$ down. Since $M_0$ adds value when deployed in advance, at time $t=0$, rather than in retrospect, at time $t=T$, this project’s value conditional on activation of $M_0$ at $t = T$ is found by setting $\gamma_0 = \sigma_0 = 0$. It is the unconditional case, $F(\tilde{P}, I)$. Hence, for $M_0$ that is deployed either at $t=0$ or not at all (i.e., at $t=T$), the (optimal) project value is:

$$\max[ F(\tilde{P}, I | M_0), F(\tilde{P}, I) ] .$$

**Two Parallel Mitigations.** Assuming stochastic project payoff $\tilde{P}$ and cost $\tilde{I}$, consider two mitigations, $M_0(\gamma_0, \sigma_0)$ and $M_1(\gamma_1, \sigma_1)$, that are both deployed at $t=0$. $M_0$ costs $c_0$ and impacts $\tilde{P}$ with impact parameters $(\gamma_0, \sigma_0)$. $M_1$ costs $c_1$ and impacts $\tilde{I}$ with impact parameters $(\gamma_1, \sigma_1)$. Adaptation of the asset-for-asset exchange model would be denoted as follows (see Appendix A.2.2 for details):

$$F(\tilde{P}, \tilde{I} | M_0 \& M_1) = Pe^{-rT+\gamma_0} \cdot p_1 - I e^{-rT+\gamma_1} \cdot p_2 - c_0 - c_1$$ \hspace{1cm} (7)$$

Here, too, the impact of $M_0$ and $M_1$ on the distributions of $P$ and $I$, respectively, makes probabilities $p_1$ and $p_2$ also a function of $(\gamma_0, \sigma_0)$ and $(\gamma_1, \sigma_1)$, boosting $p_1$ up and $p_2$ down even more than in the case of a single mitigation. And, the (optimal) project opportunity value is:

$$\max[ F(\tilde{P}, \tilde{I} | M_0 \& M_1), F(\tilde{P}, \tilde{I} | M_0), F(\tilde{P}, \tilde{I} | M_1), F(\tilde{P}, \tilde{I}) ] .$$

**Two Sequential Mitigations.** This case is more intricate because the mitigations create flexibility at two different time points. Assume a stochastic payoff $\tilde{P}$, a deterministic cost $I$, and mitigations $M_0$ and $M_1$ that cost $c_0$ and $c_1$ and can be deployed sequentially at $t_0=0$ and $t_0>0$ ($t_1<T$), respectively. Deploying $M_1$ amounts to paying $c_1$ to obtain a project opportunity that expires at time $T$ and allows acquiring $\tilde{P}$ for cost $I$. Deploying $M_0$ at cost $c_0$ yields a similar opportunity that, in addition, offers the flexibility to deploy $M_1$. On this ground, the project opportunity is treated as a compound option whose value conditional on deployment of $M_0$ and $M_1$ at $t_0=0$ and $t_1>0$, respectively, is written as (see details in Appendix A.2.3):

$$F(\tilde{P}, I | M_0, M_1) = Pe^{\gamma_0+\gamma_1} \cdot p_1 - I e^{-rT} \cdot p_2 - c_1 e^{-rT_1} \cdot p_3 - c_0$$ \hspace{1cm} (8)$$
The first term is the expected payoff $P$ conditional on: (1) the value of $P$ at $t_1$ exceeding $c_0$, the cost of $M_0$, and (2) the value of $P$ at $T$ exceeding $I$, the project cost; $p_1$ is the joint probability of both conditions being met. The second term is the expected project cost if undertaken at $T$, where $p_2$ is the probability that the project will be undertaken. The third term is the expected cost of deploying $M_1$ at $t_1$, where $p_3$ is the probability that the value of $P$ at $t_1$ will exceed $c_0$. The last term is cost of deploying $M_0$ at time 0.

To summarize, we presented the logic of option models that can account for up to two sequential mitigations. As we discuss later, for more mitigations, numeric option models are needed, not analytical models like those summarized here. Valuation aside, it is crucial to see how our SDRM solution approach is applied in practice: what inputs it requires, how it would be applied, and how to interpret its results.

5. Applying the Approach and Interpreting its Results

Let us see an example to how active risk management is applied with our approach. Given a risky system development project opportunity, deployment of three optional risk mitigations is considered. The goal is to determine which zero or more of these is worth deploying and in what sequence.

A project opportunity can be completed within half a year, or by $T=0.5$. It has a cost $I$ of $90K and expected payoff $\tilde{P}$ of $100K; both quantities are the present value of all project costs and revenues, respectively. The project’s expected net present value (NPV) is $10K, but the actual NPV can be lower or higher due to two risk factors. One is the uncertain level of customer demand for services offered by the target system; it may deviate from expectations. Another is the uncertain skills level of personnel assigned to the project; it may not fit the expected project needs. Both factors make only the payoff $\tilde{P}$ uncertain. Even the second factor is assumed to potentially impact the system quality (not cost) and, in turn, the customer demand for system services. The volatility of $\tilde{P}$, $\sigma_p$, is estimated at 40%.

The optional mitigations can be deployed independently in the sequence shown in Figure 9a. (Independence means that deployment of one mitigation does not require deploying any other mitigation.) The project could start (node $S$) directly with the development effort (node $E$), or start with deployment of the first mitigation (node $M_0$) and proceed either with development (node $E$) or with another mitigation (node $M_1$ or $M_2$), etc. Mitigation $M_0$ enables learning via a market survey that costs $c_0$: it can be deployed at $t=0$ to yield information for resolving uncertainty over customer demand. Mitigations $M_1$ and $M_2$ are mutually exclusive and can be deployed at time $t_1$. $M_1$ is a manipulation mitigation that costs $c_1$. It can improve the skill level of personnel via more training, to
relieve concerns over the quality of the system. $M_2$ allows learning via a pilot effort that costs $c_2$ and yields a payoff $P_2$ equaling 10% of $P$ starting at $t_1$. It can resolve uncertainty over customer demand as well as its sensitivity to services quality.

The interpretation of impact parameters of mitigations deserves some attention. For instance, $M_0$ can be applied at the start of the project with impact parameters $M_{0,s}(\gamma,\sigma)=(0.0,0.2)$. A mean impact of 0, $\gamma=0.0$, implies that $M_0$ does not change $\mu_P$, the expected growth rate of payoff $P$. A volatility impact of 20%, $\sigma=0.20$, implies that $M_0$ “increases” $\sigma_P$, the volatility of $P$, from 40% to 48%; recall though that this actually reflects a reduction in $\sigma_P$, consistent with the intuition offered in Figure 8. The parameters for other mitigations are interpreted in the same fashion, but note that some are path dependent. For instance, $M_1$ has two sets of impact parameters: $M_{1,0}(\gamma,\sigma)=(0.05,0.0)$ and $M_{1,s}(\gamma,\sigma)=(0.06,0.0)$. The first set implies a mean impact of 5%, $\gamma=0.05$, if $M_1$ is applied after $M_0$ has been applied, whereas the second set implies a mean impact of 6%, $\gamma=0.06$, if $M_1$ is applied without $M_0$ being applied. Likewise, $M_2$ has two sets of impact parameters: $M_{2,0}(\gamma,\sigma)=(0.0,0.16)$ and $M_{2,s}(\gamma,\sigma)=(0.0,0.18)$. The ability to handle path dependent impact parameters expands the applicability of our approach to a richer set of situations.
(a) sequencing of optional risk mitigations

![Diagram](https://example.com/diagram.png)

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project payoff (stochastic)</td>
<td>$P$</td>
<td>$100,000$</td>
</tr>
<tr>
<td>Standard deviation (volatility) of project payoff</td>
<td>$\sigma_P$</td>
<td>0.4</td>
</tr>
<tr>
<td>Project cost (deterministic)</td>
<td>$I$</td>
<td>$90,000$</td>
</tr>
<tr>
<td>Time by which project opportunity needs to be undertaken</td>
<td>$T$</td>
<td>0.5 year</td>
</tr>
<tr>
<td>Time point where mitigation $M_0$ can be deployed</td>
<td>$t_0$</td>
<td>0.0 year</td>
</tr>
<tr>
<td>Cost of mitigation $M_0$ (market research)</td>
<td>$c_0$</td>
<td>$2,100$</td>
</tr>
<tr>
<td>Mean and Volatility impact parameters of $M_0$ if applied at the start</td>
<td>$M_0,0(\gamma, \sigma)$</td>
<td>(0.00, 0.20)</td>
</tr>
<tr>
<td>Time point where mitigations $M_1$ and $M_2$ can be deployed</td>
<td>$t_1$</td>
<td>0.3 year</td>
</tr>
<tr>
<td>Cost of mitigation $M_1$ (personnel training)</td>
<td>$c_1$</td>
<td>$2,500$</td>
</tr>
<tr>
<td>Mean and Volatility impact parameters of $M_1$ if applied after $M_0$</td>
<td>$M_1,0(\gamma, \sigma)$</td>
<td>(0.05, 0.00)</td>
</tr>
<tr>
<td>Mean and Volatility impact parameters of $M_1$ if $M_0$ is not applied</td>
<td>$M_1,0(\gamma, \sigma)$</td>
<td>(0.06, 0.00)</td>
</tr>
<tr>
<td>Cost of mitigation $M_2$ (pilot effort)</td>
<td>$c_2$</td>
<td>$8,000$</td>
</tr>
<tr>
<td>Payoff from deploying mitigation $M_2$ (pilot) is 10% of $P$</td>
<td>$P_2$</td>
<td>0.1$\times P$</td>
</tr>
<tr>
<td>Mean and Volatility impact parameters of $M_2$ if applied after $M_0$</td>
<td>$M_2,0(\gamma, \sigma)$</td>
<td>(0.00, 0.16)</td>
</tr>
<tr>
<td>Mean and Volatility impact parameters of $M_2$ if $M_0$ is not applied</td>
<td>$M_2,0(\gamma, \sigma)$</td>
<td>(0.00, 0.18)</td>
</tr>
</tbody>
</table>

(b) problem parameters

Figure 9: illustrative mitigations and problem parameters

What is the best subset of mitigations to deploy? Relative to the flow diagram in Figure 9a, we must evaluate every path from node $S$ to node $E$ ($S \rightarrow E, S \rightarrow M_0 \rightarrow E, \ldots, S \rightarrow M_0 \rightarrow M_1 \rightarrow M_2 \rightarrow E, S \rightarrow M_0 \rightarrow M_2 \rightarrow E$) and pick the one that adds the most value. Each path corresponds to a compound option $F(\cdot)$ whose value is conditional on the activation of zero or more mitigations. Table 1 lists the values of $F$ computed for all paths using the compound options model for sequential mitigations (Appendix A.2.3). If no mitigation is applied, setting all impact parameters to 0% yields the value of a plain deferrable project: $F(\cdot) = 18,038$. If only $M_0$ is deployed, $F(\cdot|M_0) = 18,070$. If only $M_1$ is deployed, $F(\cdot|M_1) = 19,559$. If only $M_2$ is deployed, $F(\cdot|M_2) = 21,204$. If both $M_0$ and $M_1$ are deployed, the resulting value is: $F(\cdot|M_0,M_1) = 20,414$. And, if both $M_0$ and $M_2$ are deployed, $F(\cdot|M_0,M_2) = 21,152$. The best alternative is to deploy only $M_2$.

The results are revealing. First, that the value of a deferrable project is $18,038 means that even deferral flexibility with passive learning adds $8,038 to the plain NPV of $10,000 for an immediate project rollout. Second, when active risk mitigations are considered, the project value grows further; e.g., deploying $M_1$, which improves
personnel’s skills level, adds another $1,521. Third, the optimal strategy is to deploy only $M_2$, since it adds $3,166 to the expected project value, the most of any other combination of mitigations. Fourth, it is not worth deploying $M_0$ with $M_2$. This is surprising, but easily explainable. Deploying $M_0$ alone adds $32 while deploying it with $M_2$ drains $52 (i.e., 3,166−3,114). The reason is that the costs of $M_0$ and $M_2$ are additive but not their benefits, as both mitigations in essence target the same risk factor.

<table>
<thead>
<tr>
<th>Combinations of zero or more mitigations</th>
<th>Value of Project Opportunity</th>
<th>Net Added Value of Mitigations</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F(\cdot)$</td>
<td>$18,038$</td>
<td>$0$</td>
</tr>
<tr>
<td>$F(\cdot</td>
<td>M_0)$</td>
<td>$18,070$</td>
</tr>
<tr>
<td>$F(\cdot</td>
<td>M_1)$</td>
<td>$19,559$</td>
</tr>
<tr>
<td>$F(\cdot</td>
<td>M_2)$</td>
<td>$21,204$</td>
</tr>
<tr>
<td>$F(\cdot</td>
<td>M_0,M_1)$</td>
<td>$20,414$</td>
</tr>
<tr>
<td>$F(\cdot</td>
<td>M_0,M_2)$</td>
<td>$21,152$</td>
</tr>
</tbody>
</table>

Assumption: $\text{r}=6\%$

Table 1: Numeric evaluation results

The surprising insights from this SDRM example, which involves multiple optional mitigations, reveal the importance of conducting a systematic economic evaluation of the impacts of mitigations.

6. Discussion and Future Research
We presented an option-based approach to addressing the SDRM problem from an economic optimization perspective, and we illustrated its viability using a small realistic example. However, like with every new approach, more work is needed in three areas to make our approach usable in real-world settings. One concerns estimation of input parameters. Another relates to scalability for any number of mitigations. The last area deals with the development of support tools.

6.1. Parameter Estimation and Challenges
For our option-based approach to work in practice, attention must be paid to challenges in setting up a target SDRM problem. Once relevant risk factors and the problems they can cause are identified, respective mitigations and decision points where they may be deployed are selected, and then impact parameters for each mitigation must be estimated. The first two steps are not unique to our option-based approach and can be applied the same way they are applied today, but the last step could be challenging. The challenge is in the accurate interpretation and operationalization of the mean and volatility impact parameters. It could depend on whether the methods used to estimate impact parameters are model-based, data-based, or expert-based. These types of methods present tradeoffs worthy of exploration.

Model-based methods may be less challenging to use with firm-specific or benchmark software engineering
models. To illustrate, we use the COCOMO model (see footnote 1) to estimate the impact parameters for a mitigation aimed at controlling project cost risk. Consider a case involving uncertainty over the ‘Programmer Capability’ risk factor. Assuming all risk factors are set to ‘nominal’, the nominal effort ($E$) is 300 person-months. If project managers expect the Programmer Capability level of personnel they may be able to assign to the project to fall between ‘very low’ and ‘nominal’, the adjusted estimated effort would range between 426 and 300 person-months (see Table 2). We can now compute the mean and variance of estimated effort: for example, using the three-point Pearson-Tuckey formula, they are $\mu(E)=359$ and $\sigma(E)=21$, respectively. Suppose we can deploy a manipulation-type mitigation, $M$, to enhance programmers’ training so that the possible Programmer Capability levels that may be assigned to the project would range from ‘low’ to ‘high’. The adjusted estimated effort would then range from 357 to 255, and the re-estimated mean and variance conditional on deployment of $M$ would be $\mu(E|M)=302$ and $\sigma(E|M)=17$. These figures imply that mitigation $M$ has mean and volatility impact parameters of $\gamma(M)=15.5\%$ and $\sigma(M)=1\%$ (see Table 2). This example shows the appeal of model-based methods, but more work is needed to identify various types of software engineering models that may support the estimation of impact parameters for different types of mitigations.

Data-based estimation methods could be as useful as model-based methods, but they depend on the availability of suitable firm-specific or benchmark data. To illustrate, let us revisit the active learning example in Figure 5. This example uses historical data about the effectiveness of a learning mitigation aimed at controlling the risk of project payoff $P$. The data, a reliability matrix specified relative to the states of nature (or distribution) of $P$, was used to derive the input parameters for a decision tree model of the cash flows underlying $P$. Decision tree models are commonly used in software engineering research and practice for such purposes (Boehm, 1981; McManus, 2004). As the decision tree in Figure 5c indicates, the learning mitigation in the example does not change
P’s mean but it does change P’s standard deviation by 19% (from 104.4 to 84.8). Hence, based on the data in this example, the estimated impact parameters would be $\gamma(M)=0\%$ and $\sigma(M)=19\%$. This example highlights the relevance of data-based estimation methods, but their generalization to different types of mitigations requires more research. And, scarcity of relevant data could present a real constraint (Jones, 1998).

Expert-based estimation methods are another appealing alternative (Jørgensen, 2004; Pfleeger et al., 2005). Generally, such methods rest on human experts’ ability to calibrate previous experiences and data based on differences between previous project situations and a target project situation. For example, in our context, the Delphi method can enable the structured elicitation of expert-based judgments about mitigations’ impact parameters. Expert-based methods are of special interest in our case because they can also help to validate our approach and provide a sanity check for results it produces based on various ranges of input parameters. With this said, expert-based methods are not without weaknesses. They are subjective and may have limited visibility into the process and factors that experts consider in developing their estimates. Moreover, they are not recommended when aspects of the target reality are completely new to the experts (Pfleeger et al., 2005), as may be the case with our option-based approach to SDRM. For instance, correct and consistent interpretation of the meaning of impact parameters across experts would be crucial. And, there are always those who may criticize the continued need to rely on experts and subjective inputs, although Sullivan et al. (2001, p. 258) offer an extremely relevant counter argument:

…”the options perspective appears to be useful, because it highlights the role of flexibility in software design and gives the designer a way to think about that value as being tangible. *Software designers have to estimate such quantities as ‘likelihood of change’ today, in any case, in order to apply existing concepts such as information hiding. However, these estimates are treated without the benefit of any kind of well-grounded mathematical model.*

Beyond the tradeoffs discussed, estimating impact parameters is more challenging and worthy of future study considering an assumption made by us and most investigators in software development – independence of risk factors. In reality, risk factors may be compound: mitigating one factor may increase the probability and/or negative impact of another factor. Figure 1 recognizes this potential indirect impact of mitigations by stressing the need to re-iterate the SDRM process. Also, risk factors may be conditional on one another: one factor could cause a significant problem only if another factor occurred and caused its own problem. The challenge in such cases is not in their mere specification, as we have seen in the example in Section 5 for the case of path-dependent mitigations. Rather, the challenge is in the estimation of impact parameters. Overall, recognition of such factor dependencies is crucial to efforts to enhance and adapt our approach to real-world situations involving compound and conditional risk factors.
6.2. Scaling Up the Approach

Beyond parameter estimation issues, scaling up the approach presented requires reliance on flexible option valuation methods. Given $n$ mitigations, the goal is to maximize the value of a project opportunity by finding an optimal policy $i, t(i)$ of what sequence of mitigations to apply and at what time points. This problem could involve complicating constraints, for example: multiple mitigations may control exposure to the same risk factor, and mitigations can be path-dependent (i.e., sequences “$M_i$ followed by $M_k$” and “$M_k$ followed by $M_i$” can have different cumulative impacts). The example in Section 5 has considered both these constraints without elaborating on their computational implications.

Using analytical models to meet such constraints is just impractical, and so reliance on numeric solution methods is a must. A well-established one is that of binomial lattices, although this method would become more complex when adapted for mitigations. Koussis et al. (2004) provide one such adaptation but only for the case where project payoff $P$ is stochastic and cost $I$ is deterministic. We are working to expand their method for the more general case where both $P$ and $I$ are stochastic, by building upon the binomial version of the asset-for-asset exchange option model (Appendix A.1.2). This numeric solution method offers another crucial strength: it allows for the valuation of any given number of mitigations in one evaluation pass of the lattice. This is in sharp contrast with the way analytical options models require enumerating all subsets of zero or more mitigations, solving for each subset separately, and then taking the maximum.

Another numeric method worth examining has been developed and applied at the Boeing Corp. (Datar and Mathews, 2004). It uses Monte Carlo simulations to compute the value of an option. For instance, for a simple call option, iterations of the simulation evaluate the next expression:

$$ F = \text{if } ((P-I)>0, (P-I), 0) $$

After many iterations $F$ ends up having a truncated probability distribution and its expected value is taken as the mean of the distribution. The appeal of this method lies in three properties. First, its intuitive approach and transparency enables simplification of the mathematics involved in real option models and makes it more accessible to practicing project managers. Second, flexibility to accommodate arbitrary probability distributions for $P$ and $I$ offers a better fit with the probability distribution commonly used in software engineering research and practice; as a side point, when $P$ and $I$ are lognormal, the method is algebraically equivalent to the Black-Scholes model (Datar and Mathews, 2004). Last, Monte Carlo simulation enables generation of displays of outcome statistics that can enhance SDRM decision-making (Fairley, 1994; McManus, 2004). With this said, though, this method’s simplicity...
could also be its main weakness. It is not clear whether the method is scalable to compound and nested options of the kind used to model multiple sequential mitigations. This is another worthy direction for future research.

6.3. Support Tools and Links to Existing Work

Another area of future work is the development of software tools to facilitate use of the approach presented, similar to the way COCOMO facilitates ease of use of a complex model. In fact, one can argue that COCOMO would have not gained the widespread adoption it enjoys without tools to calibrate the model to local data and to then use the calibrated tool to make estimates. The importance of tool support has also been identified by Sullivan et al. (2001), who aptly argued that building real options-based models is a challenge especially for industrial-scale software development projects. This is also a key lesson from a recent field study that applied options-based models to a data mart consolidation project at a major airline (Benaroch et al., 2007). Any tool developed to support the approach presented must, at a minimum, include an underlying mathematical valuation model or a Monte Carlo valuation engine.

Beyond tool support, a crucial area for future research is the fitting of our approach with existing work and vice versa. In some cases this may be easy, in others not so. For example, take Bahsoon et al.’s (2005) work on the valuation of architectural stability. In line with its reliance on a standard options valuation model and the modeling of benefits from accommodating an anticipated change as a distribution (e.g., optimistic, likely, and pessimistic), this work can be cast in terms of how a more stable middleware architecture modifies the distribution of project value. And, consistent with its recognition that both a technical and a business perspective can associate different benefits with architectural potential (flexibility), our approach can cast more naturally both types of benefits in terms of their impact on the distributions of project cost and project payoff, respectively. By contrast, recasting Sullivan et al.’s (2001) work on information-hiding modularization using our approach may be more challenging, primarily because their work neither explicitly quantifies the notion of risk nor casts the value of flexibility in modular designs in terms that can be explicitly linked to business value.

Another fruitful direction is to fit our work with different agile software development methodologies, with a focus on the question: Could results of our modeling approach and their interpretations vary across software developers who follow different methodologies? The simple answer is that our approach addresses generic risk management issues that cut across methodologies. A more subtle answer must also look at development process management and governance issues. Racheva and Daneva (2008) and Racheva et al. (2008) discuss how companies (e.g., Yahoo, Microsoft) use real option concepts to prioritize requirements and manage agile development processes.
aimed at maximizing client value. Two good examples are: (1) the common practice of allowing a client to postpone her decision on the requirements until she incorporates late breaking information into her decision, and the respective practice of SCRUM and XP developers is to defer the decision about which story (Backlog items) to develop until just before coding starts; and, (2) the cancel-after-any-phase practice prioritizes work by business value and allows the customer to halt the project after any phase. These agile practices fit well real options theory: practice (1) maps to a deferral option, and practice (2) maps to an abandon option (Racheva et al., 2008). However, while real option theory may validate the intuition of agile development practices that are heuristic in nature, there is still a need to move beyond intuition and heuristic thinking. In particular, are common agile practices always on target? For instance, is passive deferral of requirements prioritization always optimal? Can proactive learning of the kind formalized by our approach add value? Likewise, is there a limit to cancel-after-any-phase scope cuts? What if a customer cancels at a given stage just to switch to a lower cost vendor or to cut scope to fit the budget? These two possibilities highlight the risk of rigidly following heuristic agile development practices. In sum, there is a need to formally examine each agile practice alone as well as how multiple practices interact when applied together. We encourage following such a line of investigation using the formal modeling approach presented.

To summarize this section, it is clear that more research is needed to turn the potential that the real option-based approach presented holds for systems development risk management into a reality that organizations can rely on in practice.

7. Conclusion
This paper presented an approach to solving the systems development risk management problem from an economic optimization perspective. Towards this end, it took three specific steps within the scope of this approach. First, fundamental problem elements, such as risk factors and risk mitigations, were formalized in terms of their monetary consequences on project value. Second, risk mitigations were identified with four generic risk management strategies, and the impacts of each strategy on distributional parameters of project value have been formalized. Last, based on these notions, standard real options models were adapted to the case of valuation of up to three sequential mitigations. Subsequently, the paper has shown that the approach presented can find what specific set of mitigations to deploy for a target project for the purpose of controlling risk while maximizing project value, and this was supported by a small but realistic example. Finally, the paper discussed several areas in which more research is needed to make the approach usable in practice.

References


Appendix A: Analytical Option Models with and without Mitigations

-------------- AVAILABLE FROM THE AUTHORS UPON REQUEST ------------