Specifying the Software Project Risk Construct

Mazen El-Masri
*HEC Montreal, Mazen.el-masri@hec.ca*

Suzanne Rivard
*HEC Montreal, suzanne.rivard@hec.ca*

Follow this and additional works at: [http://aisel.aisnet.org/amcis2010](http://aisel.aisnet.org/amcis2010)

Recommended Citation
[http://aisel.aisnet.org/amcis2010/468](http://aisel.aisnet.org/amcis2010/468)
Specifying the Software Project Risk Construct

Mazen El-Masri  
HEC Montreal  
Mazen.el-masri@hec.ca

Suzanne Rivard  
HEC Montreal  
Suzanne.rivard@hec.ca

ABSTRACT
This paper conceptualizes software project risk in terms of risk exposure and defines it as an aggregate multidimensional construct comprised of four interrelated dimensions: risk sources, risk events, risk management mechanisms, and expected outcomes. The paper also theorizes about relationships between the four dimensions. Furthermore, it argues that the risk factors identified in the literature could be systematically categorized using the proposed construct. The specification is derived from a review of 20 years of software project risk literature – from 1989 to 2009 and on a semantic decompositional analysis of software project risk definitions. The proposed construct conceptualization helps demarcate between the often intertwined behavioural factors and project attributes recognized risk factors in the literature pertaining to software project risk. By identifying the dimensions of risk and their interrelationships, the suggested specification should help improve the construct’s explanatory and predictive power.

Keywords
Information systems project risk, risk definition, risk management, risk factors.

INTRODUCTION
Software project risk has long been claimed to be a major cause of project failure (Barki et al. 1993; Boehm 1991) and empirical evidence exists to support it, with high levels of risk being associated with undesirable project outcomes such as low software quality, delays and budget overruns (Wallace et al., 2004a). Many researchers have studied software project risk, its management and its effect on project success. An examination of this research reveals that software project risk has been conceptualized in several different ways, which is reflected in different operationalizations of the construct. For instance, some authors conceptualize risk as the aggregation of risk factors (e.g. Jiang et al., 2002; Keil et al., 1998; Lu and Ma, 2004; Wallace et al., 2004a). Others refer to risk exposure, which is defined as a function of the probability that undesirable events occur and the magnitude of their associated loss (e.g.: Barki et al., 2001; Boehm, 1991; Charette, 1989; Han and Huang, 2007; Lyytinen et al., 1998). Risk has also been conceptualized as the variance from expected outcomes (Benaroch et al., 2006; Clemons, 1995; Davis, 2002).

An analysis of the various conceptualizations and of their operationalizations suggests that a clarification of the construct is warranted. First, the nature of risk factors, which are at the core of the risk factor and of the risk exposure definitions of software project risk, varies across operationalizations. Indeed, some operationalizations limit the notion of risk factors to attributes of a software project – e.g. project size, complexity, and team expertise – others also include other elements that are akin to a lack of risk management – e.g. complexity not properly managed. Second, although there is a large consensus on the idea that software project risk varies during the course of a project, the conceptualizations of risk proposed in the literature do not explicitly account for the dynamic nature of software project risk. Third, notwithstanding the general agreement that risk management mechanisms have an effect of the level of risk, the extant conceptualizations of risk do not take this effect into account.

This paper aims to propose a specification of the software project risk construct that addresses these issues. To develop this conceptualization, we reviewed 20 years of software project risk literature and we conducted a semantic decompositional analysis of software project risk definitions. From this we propose a conceptualization of software project risk exposure that comprises four interrelated dimensions: risk sources, risk events, risk management mechanisms, and expected outcomes. We also theorize about the relationships between these dimensions over the course of a project.
SOFTWARE PROJECT RISK: THE NEED FOR CONCEPTUAL CLARIFICATION


Some criteria for article exclusion were applied. First, articles that primarily investigated technical risks (such as specific data security, application security, and architectural design problems) within a project were excluded. Second, articles that dealt with project risk but that did not investigate software projects – for example, new product development projects in IEEE journals or construction projects in the Project Management Journal – were excluded. The search resulted in 87 articles that were grouped under three broad conceptualizations of software project risk: risk as risk factors, risk as risk exposure, and risk as variance of outcomes. For the sake of brevity, the 87 sources are not listed in this paper. They are, however, available upon request.

Software Project Risk as Risk Factors

The majority of authors (47 articles) conceptualize software project risk as a set of factors – that they also refer to as states, properties, aspect, events, or contingencies – that can pose a serious threat to the successful completion of a software project. Examples include lack of required knowledge (Keil et al, 1998), technical complexity (Jiang and Klien, 2000), user resistance (Moynihan, 2002), and changing scope (Schmidt et al., 2001). Factors are frequently grouped under separate dimensions like team or requirement risk. In empirical tests, researchers often compile a list of risk factors and request that practitioners evaluate their significance (e.g.: Jiang and Klein, 1999). Consequently, researchers investigate the correlation between these risk factors and project performance based on a predetermined set of performance criteria such as budget, schedule, and scope (e.g.: Jiang and Klein, 1999; Sauer et al. 2007).

Software Project Risk as Risk Exposure

The second group of authors (30 articles) define risk as risk exposure, which is a function of the probability of occurrence of an undesirable outcome and of the consequences of the occurrence of this outcome. Undesirable outcomes are defined as negative deviations from the expected project outcomes. Examples of undesirable outcomes include overall failure, budget overrun, delay, lower quality, user dissatisfaction, reliability, and performance shortfalls (e.g.: Barki et al., 2001; Han and Huang, 2007).

The assessment of risk exposure first calls for measuring the probability of occurrence of undesirable outcomes. It has been argued that probability distribution of undesirable outcomes is difficult to estimate (Barki et al., 1993; Heemstra and Kusters, 1997). Two main approaches are used to circumvent this difficulty. The first approach investigates possible risky incidents by relying on experience and prior observations (Lyytinen et al., 1998). Subsequently, it is possible to perform a root-cause analysis to identify and quantify the source of the risky incidents (Gluch, 1994). The second uses risk factors, defined as project attributes, as an estimate for the probability of occurrence of undesirable outcomes (Barki et al., 1993).

Software Project Risk as Variance of Outcomes

Ten studies conceptualized risk as variation in project outcomes. The deviation from project outcomes is predominantly described as negative. However, while these studies consider risk as negative variation from espoused outcomes, they often recognise the positive part of the failure-optimality continuum. This is generally attained by incorporating managerial behaviour like risk mitigation mechanisms to decrease a project’s risk exposure (e.g.: Benaroch et al., 2006; Kumar, 2002). Theories of economics and strategic management are predominantly employed in these studies. Consequently, outcomes are viewed as risk-adjusted project net present value commonly operationalized in monetary figures.

Risk Factors in the IS Literature

The notion of risk factor is key in the software project risk literature. Indeed, most authors who conceptualize risk as risk factors aggregate the risk factor values to quantify risk and many conceptualizations of risk as risk exposure use risk factors to estimate the probability of undesirable outcomes. Thus, the way risk factors are conceptualized in the IS literature plays a critical role in the specification of the risk construct.
A risk factor has been defined as a condition (attribute) that poses a serious threat to the successful completion of a software project (e.g.: Wallace et al., 2004b). Yet, while explicitly defined as such, undesirable events like “timetable changes”, “scope changes”, “gold plating”, and “organizational restructuring” are used as risk factors (see Ropponen and Lyytinen, 2000; Schmidt et al. 2001; Wallace et al. 2004b). Alternatively, a risk factor has also been defined as a contingency (event) presenting a threat to project success (e.g.: Keil et al., 1998). Here too, some of the risk factors used like “unskilled personnel” and “technological newness” represent attributes and not events (Keil et al., 1998). However, a project attribute such as “technological newness” at time (t₀) might result in an undesirable event like “timetable changes” at time (t₁). Accordingly, a truthful assessment of project risk requires attentiveness in regards to which risk factors to use.

DECOMPOSITIONAL ANALYSIS

The specification of the software project risk construct requires a clear definition of risk. According to Barki (2008), a clear definition is important when an attempt is made to transform a concept into a construct. However, the diverse conceptualizations of software project risk and the lack of conceptual clarity of project risk factors make it difficult to obtain a clear definition.

Accordingly, the approach taken in this paper is to arrive to a definition of software project risk by examining the existing definitions found in the IS literature. A definition derived from previously recognized ones is a first step to specify a more comprehensive yet parsimonious risk construct. A methodical and theoretically rooted technique is imperative to examine the different texts that define software project risk and lead to a clear definition.

Therefore, a process recognized as semantic decompositional analysis was chosen. According to Akmajian et al. (2001), text is interpreted by the meanings of its words (or expressions) and the relationships between them. Semantic decomposition is a scientific process that aims at developing the text’s semantic structure in order to reveal its meaning in a more precise manner. This is usually achieved by analysing the smallest units of meaning (words and expressions) that a sentence is comprised of; otherwise known as lexical items (e.g.: “project” and “Information Systems”). The analysis involves the identification of the meaning of lexical items, the relationship between their meanings, and the relationship between the meanings of the phrases within a sentence.

According to the principles of semantic decompositional analysis suggested by Akmajian et al. (2001), the identification of meaning of lexical items can be accomplished by identifying their meaning properties. Meaning properties correspond to the attributes that describe the meaning of a lexical item. Lexical items have one of these two fundamental meaning properties (Akmajian et al. 2001, p.237).

1. Meaningfulness: A description of the lexical item (e.g.: event: something that happens at a given place and time).
2. Polysemy/Ambiguity: the property polysemous or ambiguous is attributed to a lexical item when it has more than one meaning or when its meaning is unclear in a sentence (e.g.: the word “factor” signifies anything that causally contributes to some result).

On the other hand, the identification of relationships between the meanings of lexical items can be accomplished by identifying their meaning relations. A meaning relation represents the nature of semantic relationship between lexical items (e.g.: success is the opposite of failure). Akmajian et al. (2001) list three semantic relations. They are synonymy, antonymy, and hyponymy. Another semantic relation recognized by Cruse (2004 p.120) is meronymy:

1. Synonymy: words having the same meaning.
2. Antonymy: words having opposite meaning.
3. Hyponymy: words whose meanings are included in the meaning of a more general word (a kind-of relation).
4. Meronymy: words that are semantically a constituent part of other words (a part-of relation).

Finally, finding the relationships between phrases of a sentence can be accomplished by identifying the verb(s) that connect them. Generally, a sentence consists of at least two main parts, a noun phrase and a verb phrase with the verb being the most important item (Jackendoff, 1990). Verbs in a sentence could be state-verbs (e.g.: be) or action-verbs (e.g.: pose) (Jackendoff, 1990). Action verbs are useful to highlight the relationship among different concepts. This is especially true when the verb is transitive as it requires an object phrase.

Accordingly, we used the 87 identified articles and searched for explicit definitions of software project risk which resulted in 11 definitions from 15 articles (see Table 1). The other articles did not have a stated definition of risk. Semantic decomposition was performed on these definitions (see Figure 1). The first step was to decompose the identified text into lexical items which resulted in 32 items. Meaning properties were then assigned to each lexical item. However, some of the items had polysemy/ambiguity meaning properties. For example, the item “risk factor” is lexically ambiguous as it could imply anything that contributes causally to a result. Likewise, the item “risk response” was found to be polysemous as it corresponds to risk mitigation and contingency response mechanisms (Benaroch et al., 2006; Royer, 2000). As a result, the identified articles were
searched again for additional definitions or descriptions to identify the meaningfulness property of such items which brought the number of lexical items to 46. The newly identified definitions and descriptions are not included in this paper. However, they are available upon request.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Risk Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benaroch et al., 2006.</td>
<td>The downward or upward variation in expected outcomes.</td>
</tr>
<tr>
<td>Gefen et al., 2008.</td>
<td>Unforeseen contingencies related to changes and additions to the software specifications during the development period.</td>
</tr>
<tr>
<td>Keil et al., 1998.</td>
<td>A contingency that constitutes a serious threat to the successful completion of a software development project.</td>
</tr>
<tr>
<td>Wallace et al., 2004a-b.</td>
<td>A set of factors or conditions that can pose a serious threat to the successful completion of a software project.</td>
</tr>
<tr>
<td>Higuera and Haimes, 1996; Madachy, 1997.</td>
<td>A measure of the probability and severity of adverse effects.</td>
</tr>
<tr>
<td>Charette, 1989.</td>
<td>The possibility of loss with some choice involved.</td>
</tr>
<tr>
<td>Charette et al., 1997.</td>
<td>An event with a likelihood of occurrence and some potentially negative consequence.</td>
</tr>
<tr>
<td>Lyytinen et al., 1998; Ropponen and Lyytinen, 2000.</td>
<td>A state or property of a development task or environment, which, if ignored, will increase the likelihood of project failure.</td>
</tr>
<tr>
<td>Barki et al., 1993, 2001.</td>
<td>The degree of exposure to negative events and their probable consequences. A combination of the probability of an undesirable event with the magnitude of each and every foreseeable consequence.</td>
</tr>
<tr>
<td>Purao et al., 2007.</td>
<td>A particular aspect of the development task, process, or environment, which, if ignored, will increase the likelihood of project failure.</td>
</tr>
<tr>
<td>Benaroch, 2002.</td>
<td>Failure to respond to threats.</td>
</tr>
</tbody>
</table>

Table 1. Software Project Risk Definitions identified in the IS Literature
Subsequently, *meaning relations* are given to the lexical items (see Figure 2). For example, the items *possibility*, *likelihood*, and *probability* have a synonymous meaning relation because they share the same meaning. The items *factor* and *condition* have a hyponymous meaning relation because a *condition* is a kind-of a *factor* just like *event* is a kind-of a factor. Likewise, the items *importance* and *factor* have a meronymous meaning relation because *importance* is part-of the concept *factor*.

Lexical items that share synonymous and antonymous semantic properties were then grouped together and only the item that is judged to best describe its group is chosen to represent it. For example, the items *source*, *trait*, *state*, *aspect*, *condition*, *characteristic*, and *property* are synonymous and were grouped together. Subsequently, the item *source* was chosen to represent its group of items (refer to grey boxes in Figure 2). This process resulted in the identification of four dimensions of software project risk. They are risk sources, risk events, risk management mechanisms, and expected outcomes.

Additionally, the lexical items that have meronymous relations (part-of) represent the attributes of the dimension they are related to. For example, both *rate* and *degree of control* are items that have meronymous relations with *risk source* (refer to round shapes in Figure 2).

Lastly, some lexical item groups were considered redundant. These groups have hyponymous (kind-of) relations with other items that provide more inclusive meanings (refer to striped boxes in Figure 2). For example, items *successful*, *undesirable*, *negative*, *adverse*, and *failure* are all considered as kind-of a variation of expected outcomes. *Success* and *failure* are considered scale extremities while *undesirable*, *negative*, *adverse* are considered antonymous points (Akmajian et al., 2001). The identified dimensions and their corresponding attributes are explained in the following section.

---

**Figure 1. Procedure used in Semantic Decompositional Analysis**

Output

- 11 project risk definitions
- List of lexical items
- Dictionary of lexical items' meaning properties

Process

- Identify the text to be decomposed
- Deconstruct the text into lexical items
- Determine the meaning properties of lexical items
- Are all lexical items meaningful
  - Yes
    - Determine the meaning relations between lexical items
    - Group lexical items according to their meaning relations
    - Identify lexical items with meronymous meaning relations
    - Eliminate redundant lexical items and lexical item groups
  - No

Lexical items that share synonymous and antonymous semantic properties were then grouped together and only the item that is judged to best describe its group is chosen to represent it. For example, the items *possibility*, *likelihood*, and *probability* have a synonymous meaning relation because they share the same meaning. The items *factor* and *condition* have a hyponymous meaning relation because a *condition* is a kind-of a *factor* just like *event* is a kind-of a factor. Likewise, the items *importance* and *factor* have a meronymous meaning relation because *importance* is part-of the concept *factor*.

Lexical items that share synonymous and antonymous semantic properties were then grouped together and only the item that is judged to best describe its group is chosen to represent it. For example, the items *source*, *trait*, *state*, *aspect*, *condition*, *characteristic*, and *property* are synonymous and were grouped together. Subsequently, the item *source* was chosen to represent its group of items (refer to grey boxes in Figure 2). This process resulted in the identification of four dimensions of software project risk. They are risk sources, risk events, risk management mechanisms, and expected outcomes.

Additionally, the lexical items that have meronymous relations (part-of) represent the attributes of the dimension they are related to. For example, both *rate* and *degree of control* are items that have meronymous relations with *risk source* (refer to round shapes in Figure 2).

Lastly, some lexical item groups were considered redundant. These groups have hyponymous (kind-of) relations with other items that provide more inclusive meanings (refer to striped boxes in Figure 2). For example, items *successful*, *undesirable*, *negative*, *adverse*, and *failure* are all considered as kind-of a variation of expected outcomes. *Success* and *failure* are considered scale extremities while *undesirable*, *negative*, *adverse* are considered antonymous points (Akmajian et al., 2001). The identified dimensions and their corresponding attributes are explained in the following section.
Figure 2. Semantic Relations of Lexical Items
The last step was to determine the semantic relations within the sentences. As stated earlier, action verbs highlight the relationship between the different concepts. Seven action verbs that relate the various dimensions were identified (see Figure 3). They are lexical items *address, ignore, assess, avoid, recover, threaten* and *influence*. Those relationships are explained in more details in the subsequent section.

![Figure 3. Action Verbs Lexical Items (Sentence Semantic Relations)](image)

**DIMENSIONS OF SOFTWARE PROJECT RISK**

As shown in Figure 3, the semantic analysis resulted in the identification of four main dimensions of software project risk: expected outcomes, risk events, risk sources and risk management mechanisms.

**Expected Outcomes**

Software project expected outcomes refer to the targeted efficiency of the project’s process and the effectiveness of the information system to be completed (Barki et al., 2001; Nidumolu, 1996). Expected outcomes have two attributes:

1. *Magnitude* corresponds to the significance attributed to the different expected outcomes.
2. *Variance* is the degree of deviation from the targeted outcomes. While magnitude signifies the implication of not accomplishing a particular projected outcome (e.g.: the importance of system quality), variance relate to the discrepancy between the anticipated and targeted outcome.

**Risk Events**

Risk events are possible future incidents that threaten or negatively influence the successful realization of expected outcomes. Risk events are associated with one or more expected outcomes. They have two attributes:

1. *Probability* is the likelihood that a risk event will occur.
2. *Degree of control* is the extent to which the risk event, should it occur, is under the project managers’ control. A risk event is a notion of the future. Accordingly, control refers to management’s ability to act, once risk events materialize, in order to reduce deviations from expected outcomes. For example, while managers believe they can control risk events such as scope changes, they perceive events such as organizational restructuring as outside their locus of control (Wallace and Kiel, 2004).

**Risk Sources**

A risk source is an attribute relevant to the software project that may possibly cause the occurrence of one or more risk events. Risk events cannot materialize without the existence of risk sources. They have two attributes:

1. *Importance* is the score given to a risk source. It indicates the impact a risk source has on the probability of occurrence of a risk event.
2. *Degree of control* is the extent to which the risk source is under the project manager’s control. For example, unskilled staff is a relatively controllable risk source because staff could be trained to reduce its score. However, contractual constraints like tight deadlines fall outside the manager’s direct control. Generally, endogenous (internal to the project) factors are more controllable than exogenous ones (Mursu et al., 2003; Wallace and Kiel, 2004).
Risk Management Mechanisms

A risk management mechanism is a set of actions that are intended to reduce or eliminate software project risk. They correspond to both mitigation and contingency mechanisms. The main difference between mitigation and contingency mechanisms is their timing.

1. Mitigation mechanisms are actions performed prior to the possible materialization of risk events in order to reduce or eliminate the importance of risk sources (Benaroch et al., 2006).

2. Contingency mechanisms are actions that are planned to be carried out only when risk events materialize (ibid). They are intended to reduce the negative variance of expected outcomes.

Project managers plan and implement risk management mechanisms according to the strategy they select. A strategy refers to the approach taken to deal with risk. The software risk management literature recognizes six main strategies: accept, accept and control, buy information, reduce, eliminate by avoidance, and eliminate by transfer (e.g.: Boehm, 1991; Benaroch et al., 2006; Ben-David and Raz, 2001). A project manager who faces the risk that the lack of staff expertise might lead to delays could decide to mitigate the risk by training (risk reduction), hiring expertise (eliminate by avoidance), or subcontracting/outsourcing (eliminate by transfer). She could also monitor staff performance (buy information) and prepare a contingency mechanism plan (e.g.: request that staff to work overtime in case of delays) in case the project is behind the expected performance (accept and control).

A DEFINITION OF SOFTWARE PROJECT RISK

Based on the above analysis, we deduce that risk could be defined as the probability that risk sources would lead to risk events that in turn increase the negative variance from expected outcomes with predetermined magnitudes as well as the degree of which risk management mechanisms influence risk sources and the variance of expected outcomes.

Accordingly, the software project risk construct is specified as an aggregate model using four interrelated dimensions which are risk sources, risk events, risk management mechanisms, and expected outcomes. Together, all four dimensions form the risk construct and must be simultaneously present for it to exist. This is because of the relationships between the construct and its dimensions. Projects cannot be exposed to risk if no outcomes are expected or if there are no probability that risk events would materialize. Correspondingly, risk events cannot materialize without the existence of risk sources. Also, risk cannot be construed without the existence of human choice (Charette, 1989). Thus, not implementing risk management mechanisms when risk is controllable is itself a factor contributing to risk.

The aggregate model presented in Figure 4 portrays the software project risk construct and the relationship between its dimensions. Nevertheless, the relationships between the dimensions are conditional on their associated attributes. The relationship between risk sources and risk events can only be established by mapping risk source importance to risk event probabilities. Additionally, the relationship between risk source and mitigation mechanisms can only be found when risk sources are controllable. This is because the risk sources that fall outside the locus of managers cannot be controlled. On the other hand, the relationship between risk source and contingency mechanism is conditional on the materialization of risky events. Finally, contingency mechanisms are related to variance of expected outcomes and not their magnitude (Benaroch et al., 2006). The magnitudes of expected outcomes are predetermined at the outset of the project. Accordingly, contingency mechanisms cannot alter those magnitudes but instead attempt to reduce the negative deviation from those expected outcomes.
CONTRIBUTIONS AND AVENUES FOR FUTURE RESEARCH

Valuable attempts to specify the software project risk construct and establish instruments for its measurement are presented in the IS literature. While construct specification is anchored in the researcher’s approach, its merit is appraised by its predictive and explanatory power. Recently, the IS community called for disciplined specifications of constructs as they constitute the bricks used to form all theories. Software project risk has various conceptualizations and definitions which urges the need to analyze those conceptualizations and methodically examine the existing definitions.

The inspection of 20 years of pertinent literature of software project risk of different project types expands the generalizability of the construct. Semantic decompositional analysis of risk definitions and relevant text helped identify the dimensions of the risk construct and their interrelationships. As a result, the proposed conceptualization portrays the risk as an aggregate multidimensional construct formed from its four dimensions; namely: risk sources, risk events, risk management mechanisms, and undesirable outcomes.

The article offers a substantial advancement to research and practice. From a research perspective, the proposed risk construct organizes the colossal number of risk factors previously identified in the literature under four dimensions and clarifies their relationships explaining the ambiguity pertaining to what constitutes a risk factor. The project risk attributes belong to the risk source dimension; the event-type factors like requirement changes go under risk events; the managerial (in)actions go under management mechanisms; and undesirable outcomes like failed to user meet expectations fit under variations from expected outcomes. Additionally, the classification of past risk factors under the four dimensions depending on their nature helps eliminate redundant factors that were formerly aggregated under a single list, thus improving the accuracy of the project risk exposure measure.

Another contribution is the integration of risk management mechanisms as a fundamental dimension of risk. Risk management mechanisms represent the active element of risk that accentuates its temporal nature. They signify the feedback-loop between future planned management mechanisms and the current project risk state.

From a practical viewpoint, the proposed conceptualization represents a schema to better identify risk sources at project commencement. It reduces the number of risk factors to a convenient list that managers could handle. Moreover, the proposed risk exposure measurement should provide managers with a more accurate account. Owing to the broad definition of software projects and literature scope the construct is generalizable to all software project categories.
The proposed specification lays down a solid ground for future research pertaining to the interrelations between and within the software project risk dimensions. Recently, empirical evidence suggested that there exists a dynamic and complex interaction between risk factors. This interaction could cause a snowball effect during which risk evolves dynamically in unpredictable ways. The intention is to use the suggested construct as a starting point to discover those interrelationships. Our future research direction is to develop the project risk construct further. At its state, the construct delineates the four different risk constituents that are essential to assess software project risk exposure. It also exposes the basic attributes of each of the four dimensions. Those attributes offer only a basic explanation in regards to how the risk dimensions interrelate. Besides, they do not explain the interrelationships within each risk dimension. For example, the proposed construct explains that risk mitigation mechanisms apply only on controllable risk sources. Yet, it does not explain the required conditions under which mitigation mechanisms would reduce the importance of the targeted risk scores. Our next objective is to advance the construct by identifying additional attributes that are necessary to explain those interrelationships. Those attributes, if discovered, can act as the conditions for interaction.

In conclusion, the article advances the IS field by putting forward a well defined software project risk construct that is parsimonious yet generalizable and comprehensive. Parsimony owes to a concise risk construct while generalizability is due to its ability to systematically organize risk factors from different contexts that exist in the IS literature.
REFERENCES


