Journal of Enterprise Information Management

A contingency fit model of critical success factors for software development projects: A comparison of agile and traditional plan-based methodologies

Arthur Ahimbisibwe Robert Y Cavana Urs Daellenbach

A contingency fit model of critical success factors for software development projects

A comparison of agile and traditional plan-based methodologies

Received 13 August 2013
Revised 13 November 2013
Accepted 21 February 2014

Arthur Ahimbisibwe
Victoria University of Wellington, Wellington, New Zealand, and
Robert Y. Cavana and Urs Daellenbach
School of Management, Victoria University of Wellington, Wellington, New Zealand

Abstract

Purpose – While the choices available for project management methodologies have increased significantly, questions remain on whether project managers fully consider their alternatives. When project categorization systems and criteria are not logically matched with project objectives, characteristics and environment, this may provide the key reason for why many software projects are reported to fail to deliver on time, budget or do not give value to the client. The purpose of this paper is to identify and categorize critical success factors (CSFs) and develop a contingency fit model contrasting perspectives of traditional plan-based and agile methodologies.

Design/methodology/approach – By systematically reviewing the previous literature, a total of 37 CSFs for software development projects are identified from 148 articles, and then categorized into three major CSFs: organizational, team and customer factors. A contingency fit model augments this by highlighting the necessity to match project characteristics and project management methodology to these CSFs.

Findings – Within the three major categories of CSFs, individual factors are ranked based on how frequently they have been cited in previous studies, overall as well as across the two main project management methodologies (traditional, agile). Differences in these rankings as well as mixed empirical support suggest that previous research may not have adequately theorized when particular CSFs will affect project success and lend support for the hypothesized contingency model between CSFs, project characteristics and project success criteria.

Research limitations/implications – This research is conceptual and meta-analytic in its focus. A crucial task for future research should be to test the contingency fit model developed using empirical data. There is no broad consensus among researchers and practitioners in categorizing CSFs for software development projects. However, through an extensive search and analysis of the literature on CSFs for software development projects, the research provides greater clarity on the categories of CSFs and how their direct, indirect and moderated effects on project success can be modelled.

Practical implications – This study proposes a contingency fit model and contributes towards developing a theory for assessing the role of CSFs for project success. While future empirical testing of this conceptual model is essential, it provides an initial step for guiding quantitative data collection, specifies detailed empirical analysis for comparative studies, and is likely to improve clarity in debate. Since previous studies have not rigorously assessed the impact of fit between project characteristics, project environment and project management methodology on project success, additional empirically robust studies will help to clarify contradictory findings that have limited theory development for CSFs of software development projects to date.

Originality/value – Previous research for software development projects has frequently not fully incorporated contingency as moderation or contingency as fit (traditional vs agile). This research sets

Journal of Enterprise Information Management
Vol. 28 No. 1, 2015
pp. 7-33

1741-0398
DOI 10.1108/JEIM-08-2013-0060

out to develop fully a contingency fit perspective on software development project success, through contrasting traditional plan-driven and agile methodologies. To do this, the paper systematically identifies and ranks 37 CSFs for software projects from 148 journal publications and holistically categorizes them as organizational, team, customer and project factors.

Keywords Methodology, Fit, Project success, Critical success factors, Contingency, Software development projects

Paper type Research paper

1. Introduction

Today, many organizations must commit scarce and significant investments to software projects. However, numerous software projects are not delivered on time or budget and do not give value to the client (PMI, 2013b; KPMG, 2013). Shenhar (2008) reported that nearly two-thirds of software projects do not meet their time and budget goals, and often do not meet their business objectives (p. 15). Although there are many reasons proposed for why projects are not successful, numerous studies argue that software projects fail due to inappropriate choice of a project management approach (Sauser *et al.*, 2009; Murad and Cavana, 2012). Indeed, the existence of several alternative project management methodologies often makes it difficult to determine the best option (Sheffield and Lemétayer, 2013). It is also likely that users and/or software developers will tend to stick to what they are good at and will favour the project management methods with which they have had most experience (Boehm and Turner, 2004). As a result, despite the increasing range of available methodology choices, project managers are seen to frequently fail to seriously consider their alternatives (Howell *et al.*, 2010), potentially narrowly tailoring project categorization systems or using categorization criteria that are not logically linked with objectives.

Software development projects continue to fail even with the existence of communities of methodology practice such as PRINCE2, PMI and Agile that promote best practice (Standish Group, 2012). Whilst the traditional plan-driven development approaches are often regarded as too rigid to fit some environments, some project managers still try to force them to fit projects where dynamism may be crucial (Howell *et al.*, 2010). According to Wysocki's (2009) testimonial data gathered from 10,000 project managers, no more than 20 per cent of all projects have the characteristics of traditional projects, but research shows project managers continue to apply these traditional methods to projects for which they are not suited. In contrast, emergent agile methodologies promise increased customer satisfaction with lower defect rates, faster development times for solutions to rapidly changing requirements but are not well understood (Sheffield and Lemétayer, 2013, p. 462). Despite exhortation to move away from old practices, it has also been cautioned that the new methodologies are not silver bullets that guarantee success every time (Boehm and Turner, 2004). For example, Iivari and Huisman's (2007) study found that hierarchical organizations were not suitable for the deployment of agile methodologies.

Thus, it is not surprising that Tiwana and Keil's (2004) study of 720 software projects found that the use of an inappropriate methodology is actually the most critical risk driver for project failure (p. 74). Therefore, matching the project type and the software development approach would be expected to increase the chances of project success. Howell *et al.* (2010) further suggest that the lack of a decision support tool and theory connecting project types and project methodology discourages project managers from considering alternative methodologies (p. 256). This paper seeks to address this gap by identifying and categorizing critical success factors (CSFs), and

then developing a contingency fit model for contrasting traditional plan-based and agile methodologies. Thus, our research aims to answer the following research questions:

RQ1. What are the CSFs for software development projects based on the existing literature?

RQ2. What are the major categories of CSFs for traditional and agile methodologies?

In so doing, this study contributes to a body of knowledge which seeks to understand why software development projects succeed or fail, and how project success might be improved.

Unlike prior research which has largely focused on in-house information system (IS) development projects, this study takes a vendor perspective, rather than the client perspective that is mainly employed in the literature. While in-house development projects, where developers and users are members of the same organization, may be more constrained in their choice of methodology due to available skills and resources, the trend over the last decade shows an increasing tendency for firms to outsource their IS activities (Jun *et al.*, 2011), which could help to mitigate the mismatch between project characteristics and CSFs. Further, although interest in the client perspective on the CSFs related to software projects is increasing, the vendor perspective has received less attention and may give rise to additional insights or different CSFs (Taylor, 2007). The main difference in outsourcing environments is that the client and vendor share the responsibilities for managing outsourced IS projects. Some research evidence suggests that the two sides may have different perceptions of CSFs, management mechanisms, and project success, because of their different goals, structures and perspectives on coordination and risks (Sabherwal, 2003; Taylor, 2007; Jun *et al.*, 2011).

Since system vendors play a significant part and absorb considerable risk (Jun *et al.*, 2011), an integrated framework is needed for managing software development from a vendor perspective. Equally, the contingency relationships found in prior research need to be theorized further to determine how they may also apply to the study of the outsourced projects from a vendor perspective. Thus, this paper sets out to develop an integrative contingency fit framework to describe the effects of CSFs and their interaction on project success from the vendor's perspective. It is expected to advance our understanding of the CSFs of outsourced IS development projects and to provide system vendors with a set of guidelines that may be helpful for the effective management of outsourced ISD projects.

To more fully elucidate these arguments, the remainder of this paper is organized as follows: the next section compares traditional plan-based and agile software development methodologies. The relevancy of project contingency theory (PCT) in understanding software project management is then discussed. The concept of project success is examined as is the particularity of software projects. CSFs for software project success are systematically identified, ranked and categorized. Subsequently, a conceptual model is proposed and implications for research and practice elucidated.

2. Background: methodological communities of practice

Project management is composed of different vendor communities of methodology practices, each with a particular set of principles and guidelines. Some practices are

extensively developed while others are more ad hoc with members who share certain methodological commitments. These vendor communities of methodology practice can be broadly categorized as traditional plan-based and agile. Table I contrasts the two major vendor communities of project management methodology practice. These respective principles and procedures can be used as guidance for selecting and adapting a methodology that can help to achieve project success. Traditional plan-based approaches encompass PRINCE2 (Office of Government Commerce (OGC), 2009) and PMI (PMBOK, 2013), each with a set of contract-driven methodologies that seek adherence to a pre-established plan, as well as presumed certainty, stability, and ease of targeting/controlling existing processes. On the other hand, there is agile (Agile Alliance, 2001) with highly flexible methodologies that seek to embrace the changes and uncertainty involved in software development projects by remaining flexible and adaptive. While various project management methodologies with different underlying values and principles are available on the market, each presents as a credible candidate for selection and subsequent adaptation for software development projects since most projects involve a mix of the characteristics noted above. A methodology that is familiar to members of one community is also likely to be foreign and ill-understood by members of another community.

As highlighted above, the literature reveals that the selection of software development methodologies typically involves a choice from one of two broad categories; the traditional plan-based approaches and more open agile approaches (Ramesh *et al.*, 2012, p. 324). The traditional methods are essentially plan-driven approaches that follow the philosophies of PMBOK Guide (2013) or PRINCE2 manual (OGC, 2009), while agile methodologies (Agile Alliance, 2001) are less planned and assume many IS projects take place in dynamic environments, requiring projects to adapt quickly to changes (Singh *et al.*, 2012). There is a recent consensus among scholars that agility is a way of coping with external and internal changes, which are viewed as unpredictable and uncertain (Dyck and Majchrzak, 2012), with the ability to master change being a consistent theme across all agile methodology literature.

In contrast, the traditional approaches rely on what has been described as a linear or incremental life cycle (Wysocki, 2009; PMI, 2013a, b). In the linear or sequential life cycle, the project is designed to be completed in one unique cycle (Ramesh *et al.*, 2012). Each stage of the project from analysis to support is executed only once. The project moves from one stage to another sequentially when the predefined milestones or objectives are achieved. At the end of each stage, the deliverable is not the software itself but the documentation that reflects the milestones of the work undertaken (e.g. business requirements or design). The waterfall model is a well-known example of a linear model. Similarly, in this category of traditional approaches, there are also approaches based on an incremental model. In contrast to the linear model, the development phases (i.e. design, build and test) may be executed more than once (Sheffield and Lemétayer, 2013). At each stage, the scope is expanded according to a pre-specified plan. This allows phased delivery to the client (Charvat, 2003). Even though this approach allows more flexibility, it still follows a pre-determined plan developed at the beginning of the project, where adherence to that plan is expected.

Agile approaches, on the other hand, are based on an iterative or adaptive life cycle and are designed to accept and embrace change (Sheffield and Lemétayer, 2013). The iterative life cycle focuses on re-doing the project at each iteration. Therefore, at each iteration there is some learning as a result of feedback, and the next iteration might change or adapt what has been done before. While the incremental development cycle

Table I.
Major communities of project management practice: traditional and agile

Methodology practice	Estimated membership	Key attribute	Principles	Certifications	Guiding document
PRINCE2 (traditional approach)	> 250,000	Prescriptive	Seven principles defined in the PRINCE2 Manual	Two exams and no experience required	Managing Successful Projects with PRINCE2 (2009), Directing Successful Projects with PRINCE2 (2009)
PMI (traditional approach)	> 700,000	Descriptive	PMI (PMBOK Guide)	Project Management Professional certification and experience of 3-5 years of project management	A Guide to the Project Management Body of Knowledge (PMBOK Guide), 5th ed. (2013)
Agile	> 6,000 signatories	Appreciative	Defined by the Agile Manifesto	Skills are acquired by practice on agile projects not by training alone	Agile Manifesto (2001), Declaration of Interdependence (2005), personal statements of signatories

does not modify previous work (Charvat, 2003), iteration may. This is well illustrated by the agile principle of simplicity. This principle states that future features should not or need not be prepared in the current iteration as they are likely to evolve as a natural outcome of the rapid learning experienced on agile projects (Boehm and Turner, 2004). Iterative and adaptive life cycles have an advantage that arises from a continual testing throughout the project, which has a positive impact on quality (Dyck and Majchrzak, 2012). Agile methodologies suggest short iterations of less than three months and usually around four weeks (Imreh and Raisinghani, 2011). Each iteration would cover an entire development life cycle, i.e. from the requirement specifications of a particular set of functionalities to the testing and release to the client. An example of an iterative model is Scrum. In Scrum, once the scope of the sprint is approved, no additional functionality can be added (Singh *et al.*, 2012, p. 34). This means that the work done to meet the sprint is fixed, but the product backlog which contains all the features that still need to be implemented is dynamic. The latter is prioritized according to the needs of the customer. Features that deliver the most value will have a higher priority and will be developed in the next possible sprint. All the features are reprioritized as client's needs change. Table II compares traditional plan-driven and agile software development methodologies.

While debates about the superiority of one project methodology over the other continue, neither appears to be a perfect fit for all types of software development projects (Shenhar, 2001; Wysocki, 2009). According to Shenhar (2001), "one size does not fit all". Instead, project characteristics define the extent to which a particular project management methodology may be suitably applied. Wysocki (2009) identifies the key project characteristics on which success is contingent to include: levels of project risk, project complexity, project size, market stability and business value and technology type used. The level of developers' and users' experience have also been identified as other potentially significant situational factors (Jun *et al.*, 2011). These and other factors are argued to adjust the best-fit project management approach. Therefore, it is anticipated that the degree of alignment, or fit, of CSFs to project characteristics, project environment, and project management methodology combine to affect project success.

3. Theoretical background: PCT

PCT presents a body of literature that argues that not all projects are the same, and therefore they should not all be structured and managed the same way (Howell *et al.*, 2010, p. 256). The study of contingency theory in project management has gradually emerged during the last two decades with specific frameworks for project management that have been influenced by research from disciplines and fields of study like innovation, organizational theory, management, computer science, product management and engineering (Sauser *et al.* 2009, p. 667). Howell *et al.* (2010, p. 256) eloquently discuss how PCT has developed from classical organizational contingency theory building upon research from innovation (Shenhar and Dvir, 2007) and organizational (Van Donk and Molloy, 2008) perspectives of the project. Classical organizational contingency theory proposes that the effectiveness of an organization is related to its "fit" with its environment (Burns and Stalker, 1961; Lawrence and Lorsch, 1967). PCT similarly argues that the best approach to managing a project depends on context; that different conditions require different project organizational characteristics and that the effectiveness of the project is related to how well organization approaches and conditions fit each other (Howell *et al.*, 2010).

According to Sauser *et al.* (2009), a contingency approach to project management necessarily investigates the extent of fit or misfit between project characteristics and

Project parameter	Traditional software development	Agile software development
Development team	Plan oriented, adequate skills, access to external knowledge, pre-structured teams	Agile, knowledgeable, co-located and collaborative, self-organizing teams
Customers	Minimal commitment, not co-located and not empowered	Dedicated, knowledgeable, co-located, collaborative, representative and empowered
Requirements	Known early, largely stable	Largely emergent, rapid change
Architecture	Designed for current requirements	Designed for current and foreseeable requirement
Size	Larger teams	Smaller teams
Refactoring	Expensive	Inexpensive
Primary objective	High assurance	Rapid value
Fundamental assumption	Systems are fully specifiable, predictable and built through meticulous and extensive planning	High quality adaptive software developed based on principles of continuous design improvement and testing based on rapid feedback and change
Management style	Command and control	Leadership and collaboration
Knowledge management	Explicit	Tacit
Development model	Linear or incremental (anticipatory)	Evolutionary – delivery model (iterative or adaptive models)
Communication	Formal	Informal
Desired organizational form/structure	Mechanistic (bureaucratic with high formalization) aimed at large organizations	Organic (flexible and participative encouraging cooperative social action) aimed at small and medium size organizations
Quality control	Heavy planning and strict control, late, heavy testing	Continuous control of requirements, design and solutions, continuous testing
Organizational culture	Command and control	Leadership and collaborative
Market	Mature, stable	Dynamic/early markets
Measure of success	Conformance to plan	Business value delivered

Source: Adapted from Imreh and Raisinghani (2011)

Table II. Traditional plan-based vs agile software development methodology

project management approach (p. 666). This is consistent with the research examining enduring organization types drawing on contingency theory that suggests that organizational effectiveness is dependent upon the organization's ability to adapt to the environment, and that there is a need for congruence between the environment and structure (Miles and Snow, 1978). Similarly, it has often been suggested that more turbulent environments should be addressed by organic structures because coping with uncertainty is a core problem for complex organizations (Thompson, 1967).

A significant body of IS research examines risks, project success, and the relationships between the two from a contingency perspective (e.g. Nidumolu, 1996; Jiang and Klein, 2000; Barki *et al.*, 2001; Yetton *et al.*, 2000; Jiang *et al.*, 2006; Sauser *et al.*, 2009; Howell *et al.*, 2010; Jun *et al.*, 2011). These studies have argued for a contingency approach which considers project success to be dependent on how well the project as a whole is able to deal with uncertainties in the project environment. They have also provided some empirical evidence that in order to achieve project success, risk and

management strategies need to be tailored to project characteristics and objectives. However, apart from Jun *et al.* (2011) and Barki *et al.* (2001), contingency studies of software projects do not consider CSFs such as uncertainty or risk management profiles from an integrated perspective. Likewise, most of the empirical CSF studies are also limited to single management factors as a focal CSF within software development projects.

While some of the literature has investigated the CSFs for the successful implementation of a particular project management methodology or CSFs of project success independently, no previous studies have developed a comprehensive contingency fit model for comparative analysis of critical factors across both agile and traditional plan methodologies. Overall, while contingency has been argued previously for software development, empirical models have frequently not fully incorporated contingency as moderation or contingency as fit, sometimes simply modelling it as a direct effect on project success (Yetton *et al.*, 2000; Nasir and Sahibuddin, 2011). This study, therefore, seeks to address this gap by more fully developing a contingency fit model for software development project success based on the somewhat divergent perspectives of the agile and plan-driven approaches.

4. The concept of project success

A review of the project management literature reveals that the concept of project success has also been defined and measured in a range of different ways (Ika, 2009; Jugdev and Muller, 2005). This possibly arises because success criteria may differ from one project to another due to project characteristics. Ika (2009) argues that, although the concept of project success requires different approaches to its study, the idea of a universal set of project success criteria has always dominated. Pinto and Slevin (1988) had earlier acknowledged three aspects of project success concerning the implementation process, the perceived value of the project, and client satisfaction with the delivered project outcome. Shenhar *et al.* (1997) suggest two additional measures: business success and preparing for the future. However, empirical results by Lipovetsky *et al.* (1997) indicate that the importance of the latter is all but negligible. Thus, despite a well-established body of research in project management, no overall agreement on the concept of project success has emerged appropriate for all projects.

There appears to be more agreement within IS research on describing and assessing outsourced software project performance/success around the project's process and product performance (e.g. Nidumolu, 1995, 1996; Rai and Hindi, 2000; Barki *et al.*, 2001; Wallace *et al.*, 2004a, b; Jun *et al.*, 2011). Project process performance or project management success describes how well the software development process has been undertaken, measuring the extent to which a project is delivered on schedule/time, and within budget and scope (Jun *et al.*, 2011, p. 925). On-time and on-budget completion refer to the extent to which a software project meets its baseline goals for duration/schedule and cost respectively (Jun *et al.*, 2011, p. 928; Wallace, *et al.*, 2004a, b, p. 292). Software project scope refers to the work that needs to be accomplished to deliver a product, service, or result with the specified features and functions. The second dimension, project product performance, describes the performance of the system actually delivered to the users (Jun *et al.*, 2011, p. 925) and measures the quality of the resulting system. System quality, however, is a multidimensional, and also a multifaceted, concept that potentially changes over the project and product life cycle. Based on previous studies that have examined software development projects using a vendor perspective (e.g. Jun *et al.*, 2011; Wallace *et al.*, 2004a, b; Rai and Hindi, 2000),

measures for assessing the outsourced system quality address whether: the application developed is reliable; the application developed is easy to use; flexibility of the system is good; the system meets the user's intended functional requirements; the users, the project team and top management are satisfied with the system delivered; and the overall quality of the developed application is high. Understanding the range of project success dimensions is important because CSFs may well affect some success measures and not others.

5. Characteristics of software development projects

Most of the research on project management has focused on identifying CSFs for projects in industries like engineering, manufacturing and construction rather than focusing specifically on software development projects. Yet, managing software development is idiosyncratic due to the complexity, conformity, costs, flexibility and invisibility of the software itself (Nasir and Sahibuddin, 2011). Moreover, software projects have unique characteristics like code management (e.g. version control, backup, confidentiality, copyrights, etc.) and issues related to testing such as methodology, tester characteristics, time, budget, releases, etc. Unlike other engineering industries, software development projects involve complicated work of revision control which makes it possible to revert to a previous version, a critical capability for allowing editors to track each other's edits, correct mistakes and defend against vandalism and spam. In addition, the volume of data, speed of response and accuracy of expected results make the software projects relatively critical and complex (Sudhakar, 2012). Reliability, confidentiality, accountability, reportability and completeness are also crucial for software systems. Finally, software development involves many stakeholders (e.g. senior management, project manager, team members, system architects, testers, users, vendors, suppliers and customers) and each has his or her own priorities and interests that may impact on project success. Given that the combination of these characteristics could vary greatly across projects, it suggests that the importance of different CSFs will also be affected and the impact of CSFs on project success criteria may be moderated by key characteristics of software development projects (Wysocki, 2009). Such effects may in part be the reason why there is variation in the CSFs identified by different studies in the literature to date.

6. CSFs for software development projects

CSFs are issues that if addressed appropriately, substantially increase the likelihood of chances of project success (Nasir and Sahibuddin, 2011, p. 2175). However, the project management literature remains unclear about which CSFs directly affect software project success, when they impact success, and moreover, little is known about how these factors may interrelate and interact. Despite the previous research contributions in project management to identify CSFs that influence the success or failure of software projects, there is not broad agreement on these critical factors to date. Additionally, most of the research effort has focused on the outcomes of software development projects rather than the process of developing software itself. Yet, the process is argued to influence the outcomes (Nasir and Sahibuddin, 2011).

To identify CSFs, the initial step involved conducting a thorough literature search of articles in all IS project management and management journals listed on Australian Research Council (ARC) 2012 using key words such as "critical success factors", "software development" and "project success". The survey covered publications primarily from 2000 to 2012 (since agile as a term to describe methodologies came into

existence around 2001). Some of these journal articles were empirical (questionnaire and case based), while others were conceptual and experiential (written by experts and practitioners). Conference papers, book chapters, and technical papers as well as reliable web resources were included since the field of software engineering and project management evolves quickly and these outlets offer quicker publication. The identification process for CSFs involved consideration of the titles, abstracts, key sections and subsequently entire papers, because some of the factors described by the authors were not immediately clear and required careful reading to produce accurate interpretation and aggregation. The use of this approach is consistent with previous studies such as Fortune and White (2006), Nasir and Sahibuddin (2011) and Sudhakar (2012) who adopted a similar content analysis approach in deriving CSFs for project success. While these aforementioned studies also reviewed published research consisting of case studies, surveys and theoretical studies, covering different project sizes in various domains and multiple countries, the major difference is that this current research focused specifically on contrasting traditional plan-driven and agile approaches to software development projects rather than CSFs across IT projects more broadly.

In the years since the original works of identifying critical factors for software projects (Slevin and Pinto, 1987; Pinto and Prescott, 1988; Pinto and Slevin, 1988), the list of CSFs has expanded significantly. The most recent studies (e.g. Chow and Cao, 2008; Misra *et al.*, 2009; Lee and Xia, 2010; Hajjdiab *et al.*, 2012; Sheffield and Lemétayer, 2013) empirically tested CSFs of agile software development projects based on previous conceptual theorization as well as anecdotal and practical descriptions. In contrast, other authors such as Mohammad and Al-Shargabi (2011) reviewed literature based on previous research studies (such as Henderson-Sellers and Serour, 2005; Ratbe *et al.*, 2000; Cockburn, 2000; Boehm and Turner, 2003) and proposed several frameworks of CSFs also for agile projects.

All of these studies, however, are subject to some limitations. First, the more recent studies are difficult to generalize to traditional plan-driven projects because they focused primarily on the agile project context. Therefore, it would be valuable to include research on traditional plan-based projects to enable a comparative analysis of CSFs for both agile and traditional plan-driven projects. This would help guide theorization of contingencies in software development projects across the different methodologies. Second, these studies tended to rely on smaller samples ($n < 200$, with validity and reliability tests rarely reported) that are potentially restricted to specific sub-populations (individual countries, successful projects, types of respondents). These studies also do not empirically examine the direct link to project success (see Chow and Cao, 2008; Misra *et al.*, 2009; Lee and Xia, 2010 for exceptions). Others studies draw on a risk perspective and as a result may have identified similar CSFs but use different labels. Thus, although the research stream seems mature in some sense, it retains a relatively qualitative and descriptive flavour that could limit future measurement and predictive model development.

Our approach to answer *RQ1* was to survey the literature for CSFs in software development broadly. Table III shows the list CSFs for software development projects from the 148 publications identified (using the keywords noted above). Based on content analysis (Cavana *et al.*, 2001) of this extensive literature search, 37 CSFs were identified that are argued to affect project success. For each CSF identified, the frequency (number of publications) of its occurrence was counted and then expressed as a percentage of the total citation count in the literature survey ($n = 148$). Next, the identified CSFs were ranked in order of occurrence. The frequencies of CSFs for each

CSF	Total citation count in the literature ($n = 148$) Frequency	%	Agile methodology ($n = 43$) Frequency	%	Traditional plan-based methodology ($n = 105$) Frequency	%
1. Top-level management support	104	70.2	33	76.7	71	67.6
2. User/client participation	102	68.9	38	88.4	64	61.0
3. Project team commitment	98	66.2	35	81.4	63	60.0
4. Organizational culture	96	64.8	39	90.7	57	54.3
5. Level of project planning	93	62.8	6	14.0	87	82.9
6. Leadership characteristics	92	62.2	34	79.1	58	55.2
7. Vision and mission	90	60.8	6	14.0	84	80.0
8. Monitoring and controlling	88	59.5	4	9.3	84	80.0
9. Change management skills	87	58.7	36	83.7	51	48.6
10. Internal project communication	85	57.4	37	86.0	48	45.7
11. User support	84	56.8	29	67.4	55	52.4
12. Technological uncertainty	82	55.4	41	95.3	41	39.0
13. Development processes/methodologies	81	54.7	31	72.1	50	47.6
14. Technical complexity	79	53.4	38	88.4	41	39.0
15. Project team empowerment	78	52.7	35	81.4	43	41.0
16. Project team's composition	78	52.7	36	83.7	42	40.0
17. Customer training and education	78	52.7	32	74.4	46	43.8
18. Customer (client) experience	78	52.7	36	83.7	42	40.0
19. Project team's expertise with the task	77	52.0	31	72.1	46	43.8
20. Project team's general expertise	77	52.0	8	18.6	69	65.7
21. Lack of development team skill	75	50.6	28	65.1	47	44.8
22. Urgency/duration	73	49.3	24	55.8	49	46.7
23. Relative project size	73	49.2	27	62.8	46	43.8
24. Specification/requirement changes	71	47.7	34	79.1	37	35.2
25. Project team's experience with SDM	69	46.6	9	20.9	60	57.1
26. Project criticality	68	45.9	11	25.6	57	54.3
27. Lack of end user experience	67	45.2	31	72.1	36	34.3
28. Requirements and specifications	65	43.9	36	83.7	29	27.6
29. Good performance by vendors/contractors	48	32.4	4	9.3	44	41.9
30. Supporting tools and good infrastructure	45	30.4	27	62.8	18	17.1
31. Realistic schedule	43	29.1	4	9.3	39	37.1
32. Adequate resources	39	26.4	8	18.6	31	29.5
33. Risk management	37	25.0	22	51.2	15	14.3
34. Realistic budget	35	23.6	6	14.0	29	27.6
35. Good quality management	32	21.6	13	30.2	19	18.1
36. Up-to-date progress reporting	29	19.6	9	20.9	20	19.0
37. Clear assignment of roles and responsibilities	15	10.1	7	16.3	8	7.6

Note: A full list of references analysed can be obtained from the corresponding author

Table III. CSFs identified across 148 publications

software development methodology were also calculated and then plotted (see Figure 1). Studies were coded as relating to agile when this was stated as their focal context ($n = 43$). The remainder ($n = 105$), often those pre-dating 2001, were classified as relating to traditional projects. Following this, broader categories of CSFs were developed with their relative rankings of CSFs within agile and traditional studies displayed in Table IV.

While many studies have been carried out in the last 30 years to establish CSFs for software development projects, there remains only limited agreement on

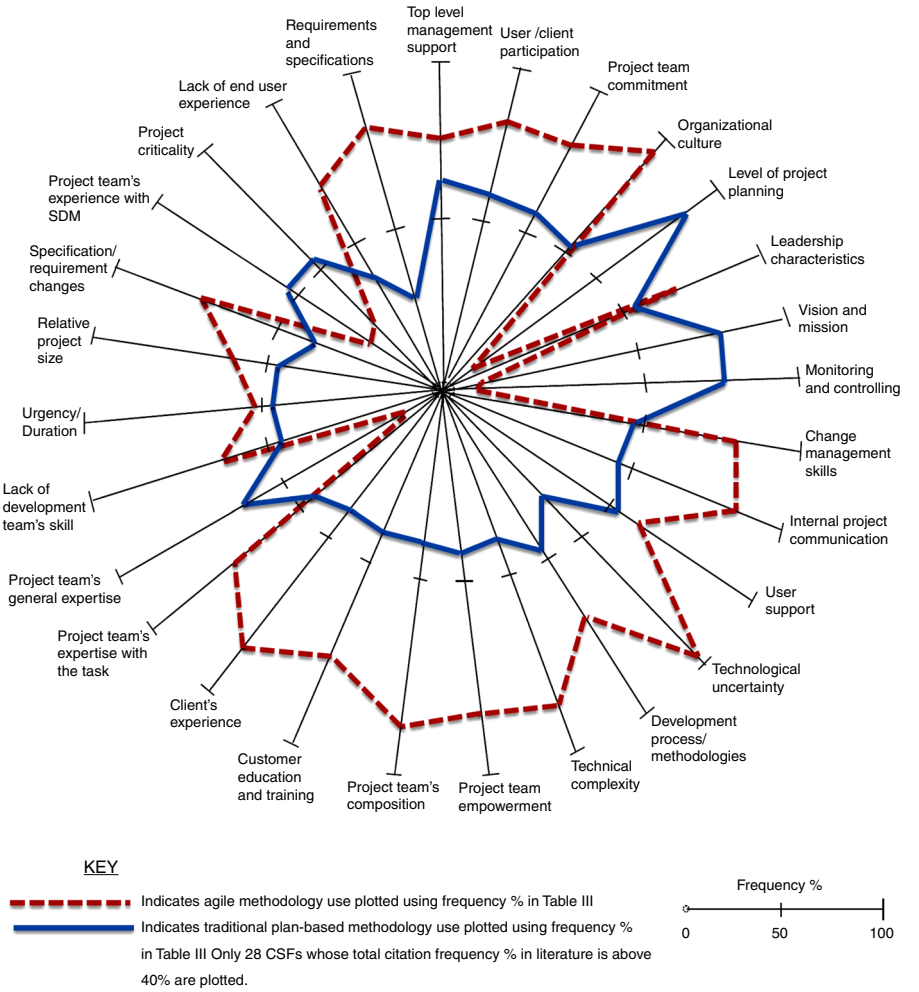

Figure 1.
CSF comparison between agile and plan-driven methodologies

what the CSFs are. Some studies have potentially created confusion by including key performance indicators (success criteria) in their list of CSFs (e.g. Oz and Sosik, 2000; Schmidt *et al.*, 2001; Sauer and Cuthbertson, 2003; Baccarini *et al.*, 2004; Charette, 2005; OGC, 2005; Standish Group, 2001).

Based on our examination, 37 distinct CSFs have been identified with the most frequently cited by about 70 per cent of the publications and 28 CSFs cited in over 40 per cent of the publications. The more frequently cited CSFs across the 148 publications tend to be factors related to top management, strategic decision making and organizational culture. Characteristics of the project teams (such as commitment, communication, composition and empowerment) are also highlighted in over 50 per cent of the publications, as are some factors associated with the customer/user.

What is also apparent in Table III and Figure 1 is that some factors are cited much more frequently in studies relating to an agile context (e.g. technological uncertainty, specification/requirement changes, organizational culture, change management,

Category	CSFs	Ranking based on number of occurrences in the considered literature (from Table III) Overall	Agile projects	Traditional plan-based projects
Organizational factors	Top-level management support	1	13	4
	Organizational culture	4	2	10
	Level of project planning	5	25	1
	Leadership	6	12	9
	Vision and mission	7	27	2
	Monitoring and controlling	8	26	3
	Change management skills	9	6	13
Team factors	Project team commitment	3	10	7
	Internal project communication	10	5	16
	Project team empowerment	15	11	21
	Project team's composition	16	7	22
	Project team's expertise with the task	19	16	19
	Project team's general expertise	20	24	5
	Lack of development team skill	21	19	17
	Project team's experience with SDM	25	23	8
Customer factors	User participation	2	3	6
	User support	11	18	12
	Customer training and education	17	14	18
	Customer (client) experience	18	8	23
	Lack of end user experience	27	17	27
Project factors	Technological uncertainty	12	1	24
	Development methodologies	13	15	14
	Project complexity	14	4	25
	Urgency	22	21	15
	Relative project size	23	20	20
	Specifications changes	24	9	26
	Project criticality	26	22	11

Table IV.
CSF categories and rankings based on methodologies

communication) whereas others appear to be of primary concern for traditional approaches (e.g. project planning, monitoring/controlling, vision and mission, team expertise, project criticality). These data clearly imply that a universal set of CSFs across all software development methodologies is unlikely and that, instead, the importance of CSFs will vary for each methodology. Some CSFs, such as user participation, user support and top management support, may be relatively important for all software development methodologies.

When considering the literature more closely, it was also apparent that many studies have grouped these CSFs within key themes (categories). However, as concluded by Fortune and White (2006), there is again no broad consensus among researchers and practitioners in categorizing CSFs for software projects. This issue remains, with recent scholars (e.g. Sudhakar, 2012; Sheffield and Lemétayer, 2013; Wan and Wang, 2010; Misra *et al.*, 2009; Chow and Cao, 2008; Mohammad and Al-Shargabi, 2011) suggesting alternative frameworks for categorizing CSFs.

By drawing on a stakeholder perspective within a software development context, though, four key themes emerged within the CSFs, each with a separate identity from

other categories. These are: organizational factors, team factors, customer factors and project factors. This categorization matches and elaborates on that utilized by others. For example, Nasir and Sahibuddin (2011) find that people factors seem to dominate the CSFs. This is not surprising because software projects more rarely fail because of technical reasons, despite the fact that people and process problems may manifest technically. It is also likely the project's technical factors can be improved with proper management of people and processes. People issues may be further elaborated on by considering which "people" associated with the project are creating the "problem" – those in the client organization commissioning the project, those in the development team or the end user or customer. This categorization also matches Sheffield and Lemétayer (2013) argument that, in order to achieve project success, the client's top management, the project team and the customer must settle on a software development approach that is aligned with the nature of the project and the environment in which it is embedded (p. 459). In other words, project stakeholders should agree and determine the appropriate software development strategy, i.e. methodology, based on project characteristics and organizational environment (CSFs) to achieve project success. Organizational CSFs, here, include all factors that are affected by top-level management, leadership, strategic direction and client organizational culture. The team CSFs relate to those employees from the client and vendor who form the development team (communication, commitment, expertise, etc.) while, the third category, customer CSFs, includes factors specific to the customer's (potentially the client organization) use of the software. The fourth and last set of factors covers project situational parameters and changes therein – corresponds to Nasir and Sahibuddin (2011) technical category.

Table IV shows a list of CSFs for software development projects identified from the literature review with their rankings based on these four major categories. Note: CSFs from Table III that addressed project success criteria are not ranked here. It should also be noted that to avoid redundancy and potential confusion, some CSFs with similar underlying meanings were merged under the most frequently cited factor. The rankings remain based on Table III to enable subsequent model building.

The rankings in Table IV suggest that not only will CSFs vary by methodology applied, but that some CSFs may be closely related, particularly those within categories, e.g. the high rankings for vision and mission, project planning, monitoring/controlling and top management support within traditional plan-based approaches is likely to indicate that the combination of these CSFs is essential for project success and that these factors may mutually support each other. In contrast, change management skills and an adaptive organizational culture may be important if the flexible methods of agile approaches are to cope with the technological uncertainty that features more prominently in these projects.

Methodology selection, therefore, should be guided by an assessment of the various CSFs identified and the conditions under which the methods are most likely to succeed. In organizational cultures where top-level management support for risk-averse attitudes is high, rigorous planning and controlling will prevail and traditional approaches are more likely (Sheffield and Lemétayer, 2013). Where the top-level management support is high for flexible cultures, changes and adaptations in budgets (costs), schedules (deadlines) occurring with agile methodologies can be more effectively accommodated. A key difference between traditional and agile approaches is the way each handles change. The traditional approach attempts to minimize change, while the agile approach embraces it. Thus, as suggested by Vinekar *et al.* (2006) traditional methodologies should be used when the future can be easily predicted while agile (adaptive and innovative) methods should be

adopted under conditions of uncertainty. Likewise, if a team lacks experience, it is more appropriate to choose a methodology with more structure and more pre-identified processes (i.e. a traditional methodology) to guide the project team members. When both the teams and customers are highly committed, knowledgeable, representative, and empowered, agile methodologies may be suitable and vice versa. With insufficient commitment and limited communication from both the project team and customers, agile projects will suffer and may fail. The literature also emphasizes that under conditions of high technological uncertainty, high technical complexity, and high specification changes, agile methodologies should be used (Charvat, 2003; Boehm and Turner, 2003). As technical complexity increases, so does the importance of process flexibility and the need to be creative and adaptive (Wysocki, 2009, p. 312).

In summary, the rankings and categorization of CSFs clearly support arguments in the literature for contingent relationships between CSFs and project factors, with software development methodology choice being a critical factor where fit is essential. The next section draws on this analysis to develop a conceptual model more formally.

7. Development of the conceptual model

As noted earlier, the focus of this study is on the selection of project management approaches that can be adopted to fit project characteristics and the project environment to manage software development projects to success. These factors (contingencies) have been grouped as organizational, team, customer and project factors.

7.1 Client's organizational factors and software development project success

Client organization factors are influences that are external to the project environment itself but are from the parent organization's context which impact on the way a project can be managed to success (Howell *et al.*, 2010). As noted, the project's environment is often dominated by the parent organization and hence the management of project is often influenced by its organizational factors. The organizational CSFs identified in the literature include top-level management support, organizational culture, project planning and controlling, leadership characteristics, change management and vision and mission. These are constructs that broadly encompass top management (leadership) strategic decisions and the inherited organizational culture. Since, software projects exist with in the broader organization; these factors can greatly impact on the management approach of such projects. Therefore, it can be logically hypothesized that different projects face different parent-imposed constraints and that that this yields different optimal project characteristics (Howell *et al.*, 2010). For instance, traditional plan-based methodologies should be used in organizations that are characterized with mechanistic and bureaucratic structures that emphasize more planning and controlling procedures while agile approaches should be used in organizations with organic and flexible structures that support more informal communication and empowerment of the project teams.

Among all organizational factors, top-level management support has been suggested to be the primary CSFs for software development projects. This is probably because top-level management commitment drives and influences other organizational factors (Jung *et al.*, 2008). Imreh and Raisinghani (2011, p. 464) and Mansor *et al.* (2011, p. 3) also emphasize that no project can finish successfully unless the project manager secures commitment from the senior management. This implies that for any project success, there is a necessity for sustained upper management commitment to provide resources, authority and influence. Consistently, Dyck and

Majchrzak (2012) found that top management commitment has a positive impact on agile software development success. Similarly, Jung *et al.* (2008) findings provided support for the hypothesis that top-level management commitment was a significant predictor of project performance. It is thus, clear that there is a positive relationship between top-level management commitment and software development project success. In regard to project management methodology selection, if the top-level management of an organization supports an adaptive behaviour, flexible leadership styles and entrepreneurial culture, they are not likely to support a rigid traditional approach that requires up-front detailed planning and formal specification, but rather agile methodologies should be adopted.

The literature also suggests that organizational culture; in particular, risk taking attitude has a positive effect on the extent to which agile projects succeed (Misra *et al.*, 2009; Sheffield and Lemétayer, 2013). Such cultures are characterized by teamwork, flexibility and participation that encourages social interaction (Strode *et al.*, 2009). Thus, software projects in these organizational environments should be managed by agile methodologies rather than the traditional plan-based approaches because agile approaches are designed to manage changes and environmental uncertainty. However, some conflicting results have been reported about the effect of organizational culture on software development project success. For instance, Misra *et al.* (2009) found that corporate culture influences success of agile projects. Similarly, Wan and Wang (2010) found that agile methods succeeded when matched with agile corporate culture. In contrast, earlier, Chow and Cao (2008) found no statistical support for the effect of organizational culture on the perceived success of agile projects.

Project planning and controlling refer to the extent to which planning and controlling practices are used in a project. Previous research has demonstrated a positive relationship between planning and process performance (Yetton *et al.*, 2000; Jun *et al.*, 2011). Poor planning is likely to be associated with inefficiencies in development and, thus, lead to large budget and time variances. Rigorously tracking and monitoring a project according to a project plan can ensure that the final product is delivered within budget and on schedule. With regard to choosing an appropriate methodology, if there is no or little change expected during the project, the plan does not need to be modified and traditional approaches that plan for every task in advance should be used. Future features should be prepared in the design and all the pieces designed to fit well together. However, since responding to change is one of the key principles of the Agile Manifesto and change usually occurs faster than plan can be updated, agile methodologies should be used when planning and controlling are not possible since they handle such situation very well.

Other organizational factors have also been found to influence software project success, e.g. Wan and Wang (2010), Sheffield and Lemétayer (2013) and Strode *et al.* (2009) found that leadership characteristics positively influence agile software development project success. This implies that if the management style has leadership characteristics and is willing to take some significant amount of risks as there is uncertainty, agile approaches should be used. In contrast, if there is a conservative environment where there is command and usually many control procedures in place, traditional approaches should be used. Similarly, Wan and Wang (2010) indicated that change management characteristics, vision and mission significantly and positively impacted software project success. Again, if the project is in highly volatile environment of change, agile methodologies should be used instead of traditional approaches since agile embrace change and uncertainty. Therefore, the client

organization's factors will tend to have a direct positive influence on project success (see Figure 2).

7.2 Team factors and software development project success

The team factors are specifically about issues of project teams and are also theorized to have a positive impact on the success of any software projects. As indicated, the success of a software project greatly depends on the team's communication, team empowerment, expertise and experience, commitment and composition. Although, these factors specifically relate to employees or project teams, some factors such as empowerment, team's composition, size and geographic distribution are also frequently influenced by the parent organization, and the broader corporate culture that is inherited from it (Howell *et al.*, 2010). This implies that the team's empowerment, composition, size, dispersal and organizational boundaries may all affect the team's ability to communicate or team's commitment. Similarly, team's communication, commitment, expertise or skill, experience and empowerment determine a team's ability to quickly comprehend and respond to the risk, thereby improving the chance of project success. Under circumstances where small teams are self-organizing, autonomous, composed of best skilled expertise and experienced, highly collaborative and committed, agile methodologies should be used and the reverse is true for traditional approaches.

Project team commitment is the willingness by a team to devote energy and loyalty to a project as expressed in three forms: affective, continuance and normative (Meyer and Allen, 1997). Affective commitment is team's emotional attachment with the

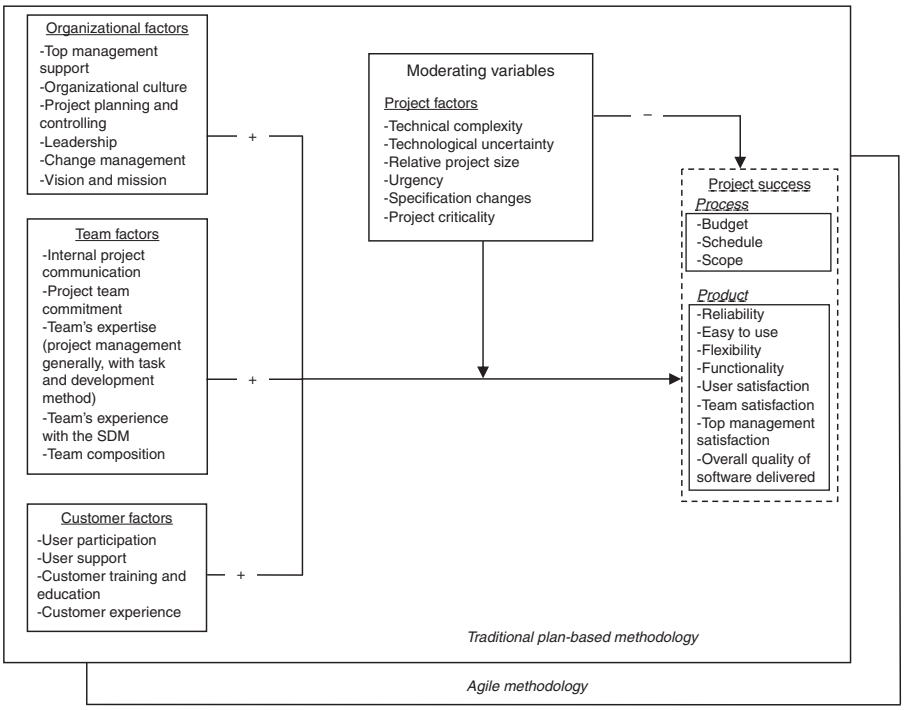

Figure 2.
A contingency fit model of CSFs for software development projects

project. Continuance commitment refers to the team's recognition of the benefits of continued association with the project compared to the perceived cost of leaving the project. Normative commitment refers to the team's feeling of obligation to remain in the project. All these three forms of commitment affect the team members' willingness to remain with a project and their work-related behaviour. Chow and Cao (2008) found that team members with great motivation positively influenced the perceived success of the agile software development projects. Correspondingly, Wan and Wang (2010) found significant positive relationships between team commitment and agile project success. This implies that committed project team members more often do not have intentions to quit, which saves the project the costs of recruiting and orienting new members in terms of both time and money. Similarly, costs of supervision are mitigated if the project team members are committed to their project tasks. If the team is highly committed, e.g. for working full time, knowledgeable, representative and empowered, then agile approaches should be used and vice versa.

Internal project communication is defined as the practices that increase information exchange and cohesion among development team members. Internal project communication enhances the levels of information sharing and collaboration between the members of the project team which decreases the amount of team conflict and keeps the team stable. Along similar lines, Jun *et al.* (2011) found that internal project communication had a significant positive effect on both process and product performance. Similarly, Yetton *et al.* (2000) demonstrated that project team conflict leads to instability in a project team and, thus, result in a project being delayed and exceeding budget. This is because software development is a knowledge-intensive and human-intensive activity that requires collaboration between team members with diverse skills and specialties. Additionally, effective internal project communication creates a feeling of responsibility and attachment between team members and the project tasks that makes team indebted to the project. As a result, this creates an atmosphere for individual team members to act without much control and coercion. Under such circumstances, what drives a person to work is the emotional attachment to the project as fostered through communication. This is consistent with the findings of Jun *et al.* (2011), Misra *et al.* (2009) and Chow and Cao (2008) who found that those workers with a positive attitude about project tasks carry out certain role behaviours well beyond the basic minimum levels required of them. They, for example, may not take extra breaks and they tend to obey the project rules and regulations even without supervisions. They attend meetings that are not mandatory but are considered important. They also keep abreast of changes within the project and elsewhere that affects or are affected by the project and responsibly discuss them with those concerned. With reference to methodology selection, if the organizational culture encourages information sharing freely and collaboration between the members with no large power distance, agile methodologies should be used and vice versa.

Other team factors have also been found to influence software project success, e.g. team capability, competences and skills have also been found to positively influence software project success (Wan and Wang, 2010; Misra *et al.*, 2009; Chow and Cao, 2008; Jiang and Klein, 2000; Boehm and Turner, 2003; Ratbe *et al.*, 2000; Lindvall *et al.*, 2002; Little, 2005). Team's expertise (general or task) includes the ability to work with uncertain objectives, ability to work with top management, ability to work effectively as a team, ability to understand human implications of a new system, and ability to carry out tasks effectively (technically) (Jiang and Klein, 2000). These are interpersonal, team or technical skills that can be determined early during the formation of the project

team. Although, these skills can be addressed from a number of viewpoints, generally, management can communicate early the basic project parameters and management guidelines to the project team to allow for skill matching. The building of team's skills can also be conducted by project managers throughout the life cycle which enhances project success. If a team's experience and expertise are low, it is more appropriate to choose a traditional methodology which provides more structure and more pre-identified processes to correctly guide the project team members. Traditional projects need highly skilled and senior staff at the beginning of the project (during the project definition phase), and then junior or lower-skilled staff can do the assigned work by following pre-established plans. On the other hand, if the talents, skills and competences of individuals in teams are high, agile approaches should be used. This is because agile projects need highly skilled and senior people throughout the entire project in order to continuously adapt to change. Thus, team factors should exhibit a direct positive influence on project success (Figure 2).

7.3 Customer factors and software development project success

The customer factors relate to the characteristics of the customers or clients of projects. In this study, customer factors are user participation and support, level of customer training and education and client experience. User participation and support consist of the behaviours and activities of the customer in relation to product development (Jun *et al.*, 2011). The literature reveals that user participation significantly increases the likelihood of the chances of software development projects success. Empirical studies by Chow and Cao (2008), Misra *et al.* (2009) and Sheffield and Lemétayer (2013) have provided data to support significant and positive relationship between user participation and support and agile software development project success. In relation to choosing an appropriate methodology, if the level of commitment of the user to participate in requirements definition and provide full-time support for the project team is high, agile approaches should be used and vice versa. This is because unlike traditional approaches, agile approaches require full time customer involvement right from the initial specification phase to the end. Although it can be argued that user participation tends to increase budget variance by encouraging suggestions for changes to specification, Yetton *et al.* (2000) found that user participation also decreases budget variance by managing expectations and quickly resolving potential problems. Similarly, Jun *et al.* (2011) also demonstrated that resolving potential conflicts early arising from greater user participation plays a vital role in the perceived system satisfaction of software developers and users. Further, customers who have an acceptable level of basic education or training in IT can easily explain their requirements and needs in a clear form. Similarly, customers who have basic knowledge about business domain accurately identify their requirements which save time, costs and contribute to process and product quality (Mohammad and Al-Shargabi, 2011). Also, customers who are knowledgeable of the exact problem of the organization to be solved are likely to help to shorten the development time in producing the product. Equally, customers who have some basic knowledge about constraints in hardware and software world can easily make choices and justify their selection of any specific hardware or software. Previous scholars have shown that end user training, experience and education play a positive role in project success (Jun *et al.*, 2011; Livermore, 2008; Misra *et al.*, 2009; Charvat, 2003; Jiang and Klein, 2000). For choosing an appropriate methodology, if the software customers are highly educated, trained or experienced in IT and are willing to participate in the development process and are supportive of the new software product, agile methodologies should be used

rather traditional approaches and vice versa. Simply because, for agile projects, the customer must be able to articulate his needs clearly, otherwise this threatens the whole project. Accordingly, customer factors are proposed to have a direct positive influence on project success (Figure 2).

7.4 Project factors (moderating variables)

In this study, "fit" refers to an alignment between project characteristics, project environment and project management methodology. It is therefore an acknowledgement that the appropriate choice of the methodology can aid the successful completion of the project. The analysis of "fit" literature broadly suggests that researchers have used various approaches to conceptualize fit. Venkatraman (1989) identified six alternative perspectives for conceptualizing and measuring fit. The six perspectives are fit as moderation, fit as mediation, fit as matching, fit as gestalts, fit as profile deviation and fit as covariation. Venkatraman's (1989) framework classifies each perspective based on three dimensions: the degree of specificity of the functional form of fit, the number of variables in the equation, and the presence of or absence of a criterion variable. According to the moderation perspective, the fit between the predictor and the moderator variable is the primary determinant of the criterion variable (Venkatraman, 1989, p. 424). The researchers usually invoke this perspective when the underlying theory specifies that the impact of the predictor varies across different levels of moderator to affect the relationship with the dependent variable (Venkatraman, 1989). Given the nature of this study, fit as moderation was found to be the most appropriate for studying project factor fit.

One of the fundamental assumptions of classical structural contingency theory is the information processing viewpoint of organizations. It is assumed that context and structure must fairly fit together if the organization is to perform well (Burns and Stalker, 1961; Lawrence and Lorsch, 1967). It is argued that organizational structures or designs enable information processing capabilities that are appropriate to the level of uncertainty challenging each organization unit (Jun *et al.*, 2011). Consequently, as the level of uncertainty facing an organization increases, decision makers must process an increasing amount of information to achieve a given level of performance. This implies that needs of an organization are better satisfied when it is properly designed and the management style is appropriate both to the tasks undertaken and the nature of the work group. Barki *et al.* (2001) general contingency hypothesis supported the view that high-risk software projects call for high information processing capacity management approaches. Similarly, based on this perspective, Jun *et al.* (2011) found that project planning and control fitted low information processing capability approaches while, internal integration and user participation represented the high information processing capability approaches to managing software project uncertainty.

Taking into consideration project types and the research perspective of this study, the key project risk factors identified from literature are: technical complexity, technological uncertainty, relative project size, and urgency, specification changes, inappropriate development methodology and project criticality. Although some of these risk factors can emerge during the course of project implementation, most of these risks are project-specific characteristics that initially exist in a project itself (Howell *et al.*, 2010). These risk management factors with different levels of inherent project uncertainty can influence different contributions of different CSFs to project success (Jun *et al.*, 2011). For instance, project planning and controlling are likely to make a greater contribution to process or product success of traditional plan-driven projects which are characterized with low levels of inherent project uncertainty than for

agile projects. Similarly, project communication is likely to make a greater contribution to process or product success of agile projects considered to have high levels of inherent project uncertainty than traditional plan-based projects with low levels. Equally, user participation and support are expected to make a greater contribution to process or product performance for agile projects featuring high levels of inherent uncertainty than traditional plan-driven projects with low levels.

Technical complexity and project uncertainty are frequently regarded as independent (e.g. Ratbe *et al.*, 2000; Shenhar and Dvir, 2007). However, authors such as Petit (2012) and Hass (2008) consider complexity and uncertainty to be aspects of the same variable. As Howell *et al.* (2010) argues, the project management issues surrounding complexity centre upon capacity to understand what is going on, and consequently predict the relationship between inputs and outputs. Lack of predictability is identical with uncertainty, and thus complexity becomes a factor in uncertainty (Howell *et al.*, 2010). Equally, use of new technologies also increases uncertainty (Howell *et al.*, 2010). Consistent with Nidumolu's (1996) study, Jun *et al.* (2011) found that the use of unfamiliar technologies can also lead to software problems that reduce the performance of the software product and delay the project. Urgency also constrains uncertainty in a similar fashion to complexity, by restraining the time resource available for understanding because decisions are made on more limited information (Howell *et al.*, 2010). Managers under time pressure also tend to take more vigorous, and often more inappropriate, measures to handle the situation thereby negatively impacting on project success.

Along similar lines, Jun *et al.* (2011) found that the absence of client knowledge and understanding of requirements or the absence of development experience and expertise within a specific application area of the development team makes it difficult to define complete, unambiguous or consistent requirements. As a result this can lead to a software product that cannot meet the client's needs, and decrease process performance. Jiang *et al.* (2006) also demonstrated that uncertainty is negatively associated with project success. Some empirical evidence reveals that project size can also affect project performance (Sauer *et al.*, 2007). Other variables such as specification changes, inappropriate development methodology and criticality also increase project uncertainty, thereby indirectly affecting project success. Specifically, Jun *et al.* (2011) established that uncertainty had a moderating effect on the relationship between planning and control, internal integration, user participation and project performance. Therefore, project factors moderate the relationship between organizational, team, customer factors and project success (Figure 2).

Project factors, such as technical complexity and technological uncertainty, can also negatively affect project success. The use of unfamiliar technologies can also lead to software problems that reduce the performance of the software product or delay the project for traditional approaches than for agile. Similarly, project criticality may demand a more plan-based approach to ensure that all project specifications are accounted for. In general, such projects are more likely to have lower process success performance since extra communication and coordination may be required. Similarly, large project size can also negatively affect project performance, more so for agile projects than traditional projects. Thus, generally, the level of project inherent uncertainty or risk associated with the project-specific CSFs is also negatively associated with both process and product success (Figure 2).

In summary, Figure 2 illustrates our conceptual model depicting the theorized relationships between all these factors and project success as well as the potential for differences between alternative methodologies.

As shown in the conceptual model, client organizational factors, team factors and customer factors are predicted to generally have a direct positive influence on project success. In contrast, project factors are hypothesized to have a moderating effect on the relationships between organizational, team and customer factors with project success as well as potentially negatively direct effects on project success. Finally, we incorporate the contingent relationships with project methodology by indicating that separate models and samples are needed for testing how the CSFs and moderated project-specific effects fit with either traditional or agile approaches.

8. Concluding remarks

This study has been carried out in four steps as follows: an extensive literature review to identify CSFs for software development projects, analysis of identified CSFs, contrasting CSFs across methodologies and developing a contingency fit model. This is the first study to develop a comprehensive contingency fit model of CSFs for software development projects, explicitly contrasting traditional plan-driven and agile methodologies. Unlike previous research, this study also categorizes project risk factors as a moderator rather than solely as an independent variable. It also fully incorporates contingency fit as potentially requiring separation model development and testing for different software development methodologies.

While the study remains conceptual and meta-analytic in its focus, it contributes towards more formally developing our theoretical frameworks for assessing CSFs for software project. This study also builds on previous research by extending the list of CSFs to 28 CSFs. We propose that methodology selection is best determined by an assessment of these 28 CSFs as well as the collective match of these CSF combinations that determine the profile of the project. Future empirical testing of this conceptual model should provide the next step, with quantitative measurement and data from sufficiently large samples enabling detailed analysis of both contingency as moderated effects and fit with a particular methodology. Overall, such testing is likely to greatly improve clarity of the debates and contrasting findings currently published in the literature.

References

Agile Alliance (2001), "Manifesto for agile software development", available at: http://agilemanifesto.org/ (accessed July 2013).

ARC (2012), *ERA 2012 Journal List*, Australian Research Council, Australian Government, Canberra, available at: www.arc.gov.au/era/era_2012/archive/era_journal_list.htm

Baccarini D., Salm G. and Love, P.E.D. (2004), "Management of risks in information technology projects", *Ind. Manag. Data Syst*, Vol. 104 No. 4, pp. 286-295.

Barki, H., Rivard, S. and Talbot, J. (2001), "An integrative contingency model of software project risk management", *Journal of Management Information Systems*, Vol. 17 No. 4, pp. 37-69.

Boehm, B. and Turner, R. (2003), "Using risk to balance agile and plan-driven methods", *IEEE Computer Society*, Vol. 36 No. 6, pp. 57-66.

Boehm, B. and Turner, R. (2004), *Balancing Agility and Discipline: A Guide for the Perplexed*, Addison-Wesley, Boston, MA.

Burns, T. and Stalker, G.M. (1961), *The Management of Innovation*, Tavistock, London.

Cavana, R.Y., Delahaye, B.L. and Sekaran, U. (2001), *Applied Business Research: Qualitative and Quantitative Methods*, John Willey & Sons, Brisbane.

Charette R.N. (2005), "Why software fails", *IEEE Spectrum*, Vol. 42 No. 9, pp. 42-49.

Charvat, J. (2003), *Project Management Methodologies: Selecting, Implementing and Supporting Methodologies and Processes for Projects*, John Wiley & Sons Inc., New York, NY.

Chow, T. and Cao, D. (2008), "A survey of critical success factors in agile software projects", *The Journal of Systems and Software*, Vol. 81 No. 6, pp. 961-971.

Cockburn, A. (2000), "Selecting a project's methodology", *IEEE Software*, Vol. 17 No. 4, pp. 64-71.

Dyck, S. and Majchrzak, T.A. (2012), "Identifying common characteristics in fundamental, integrated, and agile software development methodologies", *IEEE Computer Society. Proceedings of the 45th Hawaii International Conference on Systems Sciences, Maui*, Hawaii, *4-7 January*, available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6149536 (accessed July 2013).

Fortune, J. and White, D. (2006), "Framing of project critical success factors by a systems model", *International Journal of Project Management*, Vol. 24 No. 1, pp. 53-65.

Hajjdiab, H., Taleb, A. and Ali, J. (2012), "An industrial case study for Scrum adoption", *Journal of Software*, Vol. 70 No. 1, pp. 237-242.

Hass, K.B. (2008), "Introducing the new project complexity model", *Management Concepts*, pp. 22-31, available at: www.projecttimes.com/articles/introducing-the-new-project-complexity-model-part-i.html

Henderson-Sellers, B. and Serour, M.K. (2005), "Creating a dual-agility method: the value of method engineering", *Journal of Database Management*, Vol. 16 No. 4, pp. 1-23.

Howell, D., Windahl, C. and Seidel, R. (2010), "A project contingency framework based on uncertainty and its consequences", *International Journal of Project Management*, Vol. 28 No. 3, pp. 256-264.

Iivari, J. and Huisman, M. (2007), "The relationship between organizational culture and the deployment of systems development methodologies", *MIS Quaterly*, Vol. 31 No. 1, pp. 35-58.

Ika, L. (2009), "Project success as a topic in project management journals", *Project Management Journal*, Vol. 40 No. 4, pp. 6-19.

Imreh, R. and Raisinghani, M. (2011), "Impact of agile software development on quality within information technology organisations", *Journal of Emerging Trends in Computing and Information Science*, Vol. 10 No. 10, pp. 460-475.

Jiang, J.J., Klein, G. and Chen, H.G. (2006), "The effects of user partnering and user non-support on project performance", *Journal of the Association for Information Systems*, Vol. 7 No. 2, pp. 68-90.

Jiang. J. and Klein, G. (2000), "Software development risks to project effectiveness", *Journal of Systems and Software*, Vol. 52 No. 1, pp. 3-10.

Jugdev, K. and Muller, R. (2005), "A retrospective look at our evolving understanding of project success", *Project Management Journal*, Vol. 36 No. 4, pp. 19-31.

Jun, L., Qiuzhen, W. and Qingguo, M. (2011), "The effects of project uncertainty and risk management on is development project performance: a vendor perspective", *International Journal of Project Management*, Vol. 29 No. 7, pp. 923-933.

Jung, Y.J., Wang, J.W. and Wu, S. (2008), "Competitive strategy, TQM practice, and continuous improvement of international project management: a contingency study", *International Journal of Quality and Reliability Management*, Vol. 26 No. 2, pp. 161-183.

KPMG (2013), *Project Management Survey Report 2013*, KPMG, Wellington, available at: www.kpmg.com/NZ/en/IssuesAndInsights/ArticlesPublications/Documents/KPMG-Project-Management-Survey-2013.pdf

Lawrence, P.R. and Lorsch, J.W. (1967), "Differentiation and integration in complex organizations", *Administrative Science Quarterly*, Vol. 12 No. 1, pp. 1-47.

Lee, G. and Xia, W. (2010), "Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility", *MIS Quarterly*, Vol. 34 No. 1, pp. 87-114.

Lindvall, M., Basili, V., Boehm, B., Costa, P., *et al.* (2002), "Empirical findings in agile methods, extreme programming and agile methods", *Proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods – XP/Agile Universe*, *London*, pp. 81-92, available at: www.cs.umd.edu/~mvz/pub/agile.pdf (accessed July 2013).

Lipovetsky, S., Tishler, A., Dvir, D. and Shenhar, A. (1997), "The relative importance of project success dimensions", *R&D Management*, Vol. 27 No. 2, pp. 97-106.

Little, T. (2005), "Context-adaptive agility: managing complexity and uncertainty", *IEEE Software*, Vol. 22 No. 3, pp. 28-35.

Livermore, J.A. (2008), "Factors that significantly impact the implementation of an agile software development methodology", *Journal of Software*, Vol. 3 No. 4, pp. 31-36.

Mansor, Z., Yahya, S. and Arshad, N.H. (2011), "Towards the development success determinants charter for agile development methodology", *International Journal of Information Technology and Engineering*, Vol. 2 No. 1, pp. 1-7.

Meyer, J.P. and Allen, N.J. (1997), *Commitment in the Workplace: Theory, Research, and Application*, Sage, Thousand Oaks, CA.

Miles, R.E. and Snow, C.C. (1978), *Organizational Strategy, Structure, and Process*, McGraw-Hill, San Francisco, CA.

Misra, S.C., Kumar, V. and Kumar, U. (2009), "Identifying some important success factors in adopting agile software development practices", *Journal of Systems and Software*, Vol. 82 No. 11, pp. 1869-1890.

Mohammad, A.L. and Al-Shargabi, B. (2011), "Agile software methodologies: employee, customer and organization factors", *International Conference on Technology and Business Management*, *Dubai*, *28-30 March*, pp. 28-30, available at: www.trikal.org/ictbm11/pdf/SoftwareIssues/D1112-done.pdf (accessed August 2013).

Murad, R.S.A. and Cavana, R.Y. (2012), "Applying the viable system model to ict project management", *International Journal of Applied Systemic Studies*, Vol. 4 No. 3, pp. 186-205.

Nasir, M.H. and Sahibuddin, S. (2011), "Critical success factors for software projects: a comparative study", *Scientific Research and Essays*, Vol. 6 No. 10, pp. 2174-2186.

Nidumolu, S.R. (1995), "The effect of coordination and uncertainty on software project performance: residual performance risk as an intervening variable", *Information System Research*, Vol. 6 No. 3, pp. 191-219.

Nidumolu, S.R. (1996), "A comparison of the structural contingency and risk-based perspectives on coordination in software-development projects", *Journal of Management Information System*, Vol. 13 No. 2, pp. 77-113.

Office of Government Commerce (OGC) (2005), *Common Causes of Project Failure*, OGC, London.

Office of Government Commerce (OGC) (2009), *Managing Successful Projects with PRINCE2*, TSO, London.

Oz, E. and Sosik J.J. (2000), "Why information systems projects are abandoned: a leadership and communication theory and exploratory study", *J. Comput. Inform. Syst*, Vol. 41 No. 1, pp. 66-77.

Petit, Y. (2012), "Project portfolios in dynamic environments: organising for uncertainty", *International Journal of Project Management*, Vol. 30 No. 5, pp. 539-553.

Pinto, J. and Prescott, J. (1988), "Variations in critical success factors over the stages in the project life cycle", *Journal of Management*, Vol. 14 No. 1, pp. 5-18.

Pinto, J.K. and Slevin, D.P. (1988), "Critical success factors across the project life cycle", *Project Management Journal*, Vol. 19 No. 3, pp. 67-75.

Pinto, J.K. and Slevin, D.P. (1988), "Critical success factors across the project life cycle", *Project Management Journal*, Vol. 19 No. 3, pp. 67-75.

PMI (2013a), *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*, 5th ed., Project Management Institute, Newtown Square, PA.

PMI (2013b), *Pulse of the Profession Report 2013*, Project Management Institute Inc., Newtown Square, PA, available at: www.pmi.org/Knowledge-Center/Pulse/~/media/PDF/Business-Solutions/PMI-Pulse%20Report-2013Mar4.ashx

Rai, A. and Hindi, A.H. (2000), "The effects of development process modeling and task uncertainty on development quality performance", *Information & Management*, Vol. 37 No. 6, pp. 335-346.

Ramesh, B., Mohan, K. and Cao, L. (2012), "Ambidexiterity in agile distributed development: an empirical investigation", *Information System Research*, Vol. 23 No. 2, pp. 323-339.

Ratbe, D., King, W.R. and Kim, Y.G. (2000), "The fit between project characteristics and application development methodologies: a contingency approach", *Journal of Computer Information Systems*, Vol. 40 No. 2, pp. 26-33.

Sabherwal, R. (2003), "The evolution of coordination in outsourced software development projects: a comparison of client and vendor perspectives", *Information and Organization*, Vol. 13 No. 3, pp. 153-202.

Sauer, C. and Cuthbertson, C. (2003), "The state of IT project management in the UK 2002-2003", Computer Weekly, 15 April, pp. 1-82.

Sauer, C., Gemino, A. and Reich, B.H. (2007), "The impact of size and volatility on IT project performance: studying the factors influencing project risk", *Communications of the ACM*, Vol. 50 No. 11, pp. 79-84.

Sauser, B.J., Reilly, R.R. and Shenhar, A.J. (2009), "Why projects fail? How contingency theory can provide new insights – a comparative analysis of NASA's Mars climate orbiter loss", *International Journal of Project Management*, Vol. 27 No. 7, pp. 665-679.

Schmidt, R., Lyytinen, K., Keil, M. and Cule, P. (2001), "Identifying software project risks: an international delphi study", *J. Manage. Inform. Syst*, Vol. 17 No. 4, pp. 5-36.

Sheffield, J. and Lemétayer, J. (2013), "Factors associated with the software development agility of successful projects", *International Journal of Project Management*, Vol. 31 No. 3, pp. 459-472.

Shenhar, A. (2008), "Unleashing the power of project management", *Industrial Management*, Vol. 50 No. 1, pp. 14-18.

Shenhar, A.J. (2001), "One size does not fit all projects: exploring classical contingency domains", *Management Science*, Vol. 47 No. 3, pp. 394-414.

Shenhar, A. and Dvir, D. (2007). *Reinventing Project Management: The Diamond Approach to Successful Growth and Innovation*, Harvard Business School Press, Boston, MA.

Shenhar, A.J., Levy, O., and Dvir, D. (1997), "Mapping the dimensions of project success", *Project Management Journal*, Vol. 28 No. 2, pp. 5-13.

Singh, A., Singh, K. and Sharma, N. (2012), "Managing knowledge in agile software development", available at: http://research.ijcaonline.org/irafit/number9/irafit1072.pdf (accessed July 2013).

Slevin, D. and Pinto, J. (1987), "Balancing strategy and tactics in project implementation", *Sloan Management Review*, Vol. 29 No. 1, pp. 33-41.

Strode, D.E., Huff, S.H. and Tretiakov, A. (2009), "The impact of organizational culture on agile method use", *42nd Hawaii International Conference on System Sciences*, *HICSS*, *Hawaii*, pp. 1-9.

Standish Group International Inc. (2001), "Extreme Chaos", technical report, Standish Group International Inc., Boston, MA.

Standish Group International Inc. (2012), "CHAOS 2012 report agile projects successful 3X more than non-agile projects", Standish Group International Inc, Boston, MA, available at: www.sds-consulting.com/blog/agile-projects-successful-3x-more-non-agile-projects (accessed August 2013).

Sudhakar, P. (2012), "A model of critical success factors for software development", *Journal of Enterprise Information Management*, Vol. 25 No. 6, pp. 537-558.

Taylor, H. (2007), "Outsourced IT projects from the vendor perspective: different goals, different risks", *Journal of Global Information Management*, Vol. 15 No. 2, pp. 1-27.

Thompson, J.D. (1967), *Organizations in Action*, McGraw-Hill, New York, NY.

Tiwana, A. and Keil, M. (2004), "The one-minute risk assessment tool", *Communications of the ACM*, Vol. 47 No. 11, pp. 73-77.

Van Donk, D.P. and Molloy, E. (2008), "From organising as projects to projects as organizations", *International Journal of Project Management*, Vol. 26 No. 2, pp. 129-137.

Venkatraman, N. (1989), "The concept of fit in strategy research: toward a verbal and statistical correspondence", *Academy of Management Review*, Vol. 14 No. 3, pp. 423-444.

Vinekar, V., Slinkman, C.W. and Nerur, S. (2006), "Can agile and traditional systems development approaches coexist? An ambidextrous view", *Information Systems Management*, Vol. 23 No. 3, pp. 31-42.

Wallace, L., Keil, M. and Rai, A. (2004a), "Understanding software project risk: a cluster analysis", *Information & Management*, Vol. 42 No. 1, pp. 115-125.

Wallace, L., Keil, M. and Rai, A. (2004b), "How software project risk affects project performance: an investigation of the dimensions of risk and exploratory model", *Decision Sciences*, Vol. 35 No. 2, pp. 289-321.

Wan, J. and Wang, R. (2010), "Emperical research on critical success factors of agile software process improvement", *Software Engineering and Applications*, Vol. 3 No. 12, pp. 1131-1140.

Wysocki, R.K. (2009), *Effective Project Management: Traditional, Agile, Extreme*, Wiley, Indianapolis, IN.

Yetton, P., Martin, A., Sharma, R. and Johnston, K. (2000), "A model of information systems development project performance", *Information Systems Journal*, Vol. 10 No. 4, pp. 263-289.

Further reading

Boehm, B. and Turner, R. (2005), "Management challenges to implementing agile processes in traditional development organizations", *IEEE Software*, Vol. 22 No. 5, pp. 30-39.

Ceschi, M., Sillitti, A., Succi, G. and De Panfilis, S. (2005), "Project management in plan- based and agile companies", *IEEE Software*, Vol. 22 No. 3, pp. 21-27.

Cooke-Davies, T. (2002), "The 'real' success factors of projects", *International Journal of Project Management*, Vol. 20 No. 3, pp. 185-190.

Dybå, T. and Dingsøyr, T. (2009), "What do we know about agile software development?", *IEEE Software*, Vol. 26 No. 5, pp. 6-9.

Mishra, D. and Mishra, A. (2011), "Complex software project development: agile methods adoption", *Journal of Software Maintenance and Evolution*, Vol. 23 No. 8, pp. 549-564.

Perminova, O., Gustafsson, M. and Wikström, K. (2008), "Defining uncertainty in projects: a new perspective", *International Journal of Project Management*, Vol. 26 No. 1, pp. 73-79.

Ward, S. and Chapman, C.B. (2003), "Transforming project risk management into project uncertainty management", *International Journal of Project Management*, Vol. 21 No. 2, pp. 97-105.

Yeo, K.T. (2002), "Critical failure factors in information system projects", *International Journal of Project Management*, Vol. 20 No. 3, pp. 241-246.

Corresponding author
Dr Arthur Ahimbisibwe can be contacted at: arthur.ahimbisibwe@vuw.ac.nz