Controlling Software Project Risks – an Empirical Study of Methods used by Experienced Project Managers.

TOM ADDISON
University of the Witwatersrand
and

SEEMA VALLABH
KPMG

The failure rate of software projects has been proven to be very high, and the incidence of failure is becoming worse as more companies venture into software development. Risk management is a collection of methods aimed at minimising or reducing the effects of project failure. This research report has focused on experienced project manager's perceptions of software project risks and controls. It reports on the more significant risks and controls that are utilised to reduce the occurrence of the risk factors, or minimise the impact of various risks. Risk factors involved in software projects along with controls to mitigate these factors were identified in the literature. These were then used in an empirical study to determine their importance and frequency of occurrence. The effectiveness of various controls to reduce the occurrence of risk factors was also identified and discussed. Experienced project managers were found to use certain controls more than inexperienced project managers, particularly 'assign responsibilities to team members' and 'stabilise requirements and specifications'.

Categories and Subject Descriptors: K.6.1 **[Management of Computing and Information Systems**]: Project and People Management - *Management Techniques*; K.6.3. [**Management of Computing and Information Systems**]: Software Management;.

General Terms: Management, Human Factors

Additional Key Words and Phrases: Risks, Control methods, Management.

1. RISK AND RISK MANAGEMENT

As business reliance on software grows, so do the business-related consequences of software failure. Most software projects take place in an unpredictable environment in which many pitfalls exist that may affect the successful outcome of a project. Post-project evaluations reveal that many of the problems encountered were in fact predictable.

A project is usually deemed as successful if it meets requirements (of measures such as functionality, reliability, maintainability, portability, efficiency, integration and operability) is delivered on time and delivered within budget [Powell and Klein, 1996]. May [1998], citing a 1995 Standish survey, advises that only one-sixth of all projects were completed on time and within budget, one-third of all projects were cancelled and over half were considered 'challenged'. Keil, Cule and Lyytinen [1998] assert that the high failure rate is due to managers not taking cautious measures to assess and manage the risks involved in software projects. A study conducted by Ewusi-Mensah and Przasnyski [1991] showed that 35% of abandoned projects are not abandoned until the implementation stage of the project's life cycle. This suggests that project managers are doing a poor job of identifying or terminating projects that are likely to fail. While there are many different modes of failure, one that has occurred very often is the project that takes on 'a life of its own'. It continues to absorb valuable resources without ever reaching its goal [Keil, 1995]. Eventually these projects are abandoned but the cost of having funded them can result in a loss of organisational resources.

Software risk management is an approach that attempts to formalise risk oriented correlates of development success into a readily applicable set of principles and practices [Ropponen and Lyytinen, 2000]. It incorporates techniques and guidelines to identify, analyse and control software risk. Risk management is aimed at taking counter measures to either prevent risks from affecting the project or to reduce their impact [Heemstra and Kusters, 1996], and should be viewed as a fundamental component of the project management process [Powell and Klein, 1996]. Ropponen and Lyytinen [2000] believe that by including risk management in a project the exposure to software risk can be reduced and can thereby increase software quality and improve software development.

Author Addresses:
T.M. Addison, School of Economic and Business Sciences, University of the Witwatersrand, Private Bag 3, Wits, 2050, South Africa; tom@isys.wits.ac.za
S. Vallabh, KPMG, Private Bag 9, Parkview, 2122, South Africa; seema.vallabh@kpmg.co.za

A program of risk analysis and abatement can reduce the probability of major software disasters and also shorten average development cycles at the same time. In risk management, success is not always guaranteed but it does have the goal of identifying and responding to potential problems with sufficient time to avoid crisis situations [Heemstra and Kusters, 1996]. With this in mind, it is possible for project managers to achieve the main goal of project management i.e. control of risks.

> 'While we can never predict the future with certainty, we can apply structured risk management practices to peek over the horizon at the traps that might be looming, and take actions to minimize the likelihood or impact of these potential problems' [Cule, Schmidt, Lyytinen and Keil, 2000].

Much work has been done on identification of risk factors for project appraisal purposes. Phelps [1996] however states that much less attention has been paid to the process of managing risks as they arise during a project. It is known that developing software is risky, yet very few organisations perform any formal risk assessment to see if the risks can be controlled, reduced or managed. Project risk assessment assists the practitioner in prioritising the search for ways to counteract the difficulties that may exist. Symptoms that an organization is not effectively practising risk management include a continual state of project instability, constant fire-fighting, multiple schedule slippages because of recurring surprise factors, and constantly operating in a high-stress, crisis management role [Cule et al., 2000]. Formal risk management greatly improves the likelihood of successful project completion, and it reduces the potential negative consequences of those risks that cannot be avoided.

2. RESEARCH OBJECTIVES

The understanding of the factors that contribute to software project risk is becoming increasingly important. This is a result of the size, complexity and strategic importance of many of the information systems currently being developed. Studies by Boehm [1991] and Phan et al. [1998] show that most projects fail managerially, not technologically. This research attempts to identify and understand the factors that contribute to software project risk and to identify the process of controlling software risks.

The objectives of this study were to:

— Identify the risks involved in a software project;
— Rank them in order of importance and frequency of occurrence; and
— Identify the activities performed by project managers to control the risks identified.

3. SOFTWARE PROJECT RISKS

The risks listed below have been identified by many researchers and regularly occur in software projects. These risks are considered to be issues that need to be addressed and thereafter controlled. Each risk will be discussed in turn.

3.1 Unclear or misunderstood scope/objectives

Boehm and Ross [1989] point out that the different stakeholders in a software development project have their own objectives, which often conflict with the objectives of another stakeholder. For instance, users require a robust, user-friendly system with many functions that can support their tasks while development team members on the other hand hope to encounter interesting technical challenges. These differing expectations, according to Boehm and Ross [1989] create fundamental conflicts when simultaneously approached, resulting in unclear or misunderstood scope/objectives of the project.

3.2 Unrealistic schedules and budgets

Abdel-Hamid, Sengupta & Swett [1999] stated that the 'scheduling and timing' risk is the major complicating factor as it is difficult to estimate schedules with acceptable accuracy and consistency. Very often, organisations embark on a large project having underestimated its size and complexity. This risk leads to the difficulties in scheduling the project correctly. Ropponen and Lyytinen [2000] believe that performance with scheduling and timing risks improves with project experience.

A fixed schedule may lead to schedule pressures and people under pressure do not necessarily work better, resulting in the inability to produce satisfactory results or deliver any software at all.

3.3 Lack of senior management commitment to the project

Keil et al. [1998] conducted a study where experienced software project managers identified and ranked the most important risks. Their findings were that lack of senior management commitment was seen as the most critical risk. Gioia [1996] states that a project succeeds only when senior management makes it a top priority as management support influences a project's process and progress.

3.4 Failure to gain user involvement

This risk was also fundamental on the list identified by Keil et al. [1998]. If users are not involved, there is the risk that developers may assume detailed functionality and business requirements, leading to project objectives not being achieved. It also means laying the blame for 'lack of client responsibility' on the project manager rather than on the users [Mursu et al., 1996].

3.5 Inadequate knowledge/skills

Project personnel may not have adequate knowledge of the technology, or the business, or may just not have the experience to handle the project [Keil et al., 1998].

McLeod and Smith [1996] indicate that 'people' risk arises from inadequate skills (both technical and managerial) as well as level of experience.

The lack of experience with technology also increases the likelihood of this risk occurring [McFarlan, 1981]. Unrealistic expectations of personnel's abilities may affect the project's outcome.

3.6 Lack of effective project management methodology

Those software projects that are implemented and do reach the operational mode are often poorly rated by the users. According to Martin et al. [1994], the main reason for this is the developers not using a good methodology. On the other hand, Cash, McFarlan and McKenny, [1988, pg 13] state that using a single methodology for several years can create a straitjacket environment. The approach will normally fit one kind of project very well and others considerably less.

3.7 Misunderstanding the requirements

It may be time consuming and difficult to collect and record all of the required details from all prospective users, resulting in the project team not knowing enough about what is required to complete the project successfully [Martin et al., 1994, pg 322]. This may lead to the possibility of developing a system that cannot be used mainly because a proper systems analysis to develop a complete and accurate set of requirements has not been performed [Mursu et al., 1996].

3.8 Gold plating

Often developers and analysts think of additional capabilities or changes, known as gold plating, which they think would make the system better and more attractive in their view. These deviations may result in unsatisfied users and unnecessary costs [Lubelczyk and Parra, 1992].

3.9 Continuous requirement changes

A 'continuous stream of requirement changes' was identified by Boehm [1991], as a significant risk. As the users' needs change, so do the requirements of the project. Mursu et al. [1996] suggested that by freezing a part of the functionality and delivery date, completion of the system is enabled. Keil et al. [1998] on the other hand argued that requirements should not be frozen because in today's fast moving business environment, a frozen design does not accommodate changes in the business practices. With a frozen design, the developer has little flexibility in changing the specifications. Continuous and uncontrolled changes in requirements, however will inevitably lead to a delay in the project schedule.

3.10 Developing the wrong software functions

Boehm [1991] also established in his 'top ten risks' list, the task of getting the system functionality correct from both the user and technical points of view. Ropponen and Lyytinen [2000] suggest this risk is less likely to arise when project managers have higher experience and level of education. This risk occurs for the reason that users make changes or improvements to incorporate new factors that may arise and this may not be communicated well to the project team.

3.11 Subcontracting

Ropponen and Lyytinen [2000] identified this risk to be important because of the consequences in managing contracted tasks and the shortfalls in externally developed components.

3.12 Resource usage and performance

Incorrect evaluation of performance requirements can result in an inability to implement the system as a result of inappropriate technical solutions and computing power [Ropponen and Lyytinen, 2000]. Larger organisations tend to experience resource usage and performance risk more often since they develop complex systems that can lead to poor performance of the resulting system [Ropponen and Lyytinen, 2000].

3.13 Introduction of new technology

This risk occurs by using new or 'leading edge', technology that has not been used successfully at other companies. This risk may also be increased further if there is a shift in technology during the project [Mursu et al., 1996].

3.14 Failure to manage end user expectations

Expectations determine the success or failure of a project. If expectations are mismatched with the deliverable, problems are created [Keil et al., 1998]. Expectations should be correctly identified and constantly reinforced in order to avoid failure.

4. SOFTWARE PROJECT CONTROLS

Software project management has become a critical task in many organisations. With new risks occurring whenever a project is implemented, there needs to be a formal method of controlling them. Abdel-Hamid et al. [1993] assert that a control system for software projects needs a measuring method to detect the activities performed; a mechanism for comparing activities performed with a standard or expectation of what should be carried out; a procedure for changing behaviour if there is a need; and a feedback method or mechanism.

If control systems such as this are not in place, people are inclined to report favourable information and to withhold unfavourable information [Abdel-Hamid et al., 1993; Lauer, 1996]. Having to cope with such hurdles has frustrated software practitioners and researchers.

> 'Development efforts have been cancelled not for inadequate programming or for lack of technology but from the sheer frustration of all concerned in attempting to determine when, if ever, they would be completed.' [Lehman, 1979]

The controls below have been identified in the existing literature on risk management, (as referenced below) and are considered important in reducing the risks identified.

4.1 Develop and adhere to a software development plan

Lubelczyk and Parra [1992] suggest that a list of all known potential and relevant risks should be included as a section in all software project plans. As an alternative, individual risk factors can be documented in more detail [Cule et al., 2000]. Project control processes should also record updates to risks and their approach to key milestones. A prioritisation mechanism can be used to learn where to focus risk control energy. Cule et al. [2000] suggest that goals should be set for determining when each risk item has been satisfactorily controlled. Each mitigation action should be assigned to an individual for implementation and monitoring of effectiveness, together with a target date for completion. The most visible top ten risks should be maintained and regularly tracked for effectiveness of the mitigation approaches [Cule et al., 2000].

4.2 Combine internal evaluations with external reviews

Internal evaluations combined with external reviews should be performed to keep the project on the right track [Cule et al., 2000]. This control applies to risks of inappropriate or insufficient staffing, lack of effective development process methodology, poor estimation and improper definition of roles and responsibilities [Keil et al., 1998]. This step should include reviews of the strategic business plan of the company as well as a technology assessment. Along with the external analysis, a review of the internal strengths and weaknesses of the IS department should also be conducted to keep the project on the right track [Martin et al., 1994, pg 510]. The key is to recognize the risk areas early enough so that project managers can take appropriate preventative actions [Cule et al., 2000].

4.3 Involve management during the entire project lifecycle

Both IS senior management and project managers should be able to create and maintain long-term relationships with users and promote user commitment to the project. Project managers should periodically determine the level of commitment from senior management and the user community so that support for the project does not evaporate.

Boehm and Ross [1989] posit that the primary job of the software project manager is to structure the project to meet the 'win' conditions of various stakeholders.

4.4 Involve users during the entire project lifecycle

Managers should not expand the scope of the project without consulting the users and revising the project budget and schedule [Martin et al., 1994, pg 340].

4.5 Ensure there is a steering committee in place

An executive steering committee can be an effective means to ensure that management processes are established and maintained [Martin et al., 1994, pg 610 – 611]. To be effective, the steering committee should therefore have the ultimate authority on the approval of projects. Regular meetings should be held to report on the status of ongoing projects [Robert, 1997] and communicate concerns, issues and possible remedies [Martin et al., 1994, pg 610 – 611].

4.6 Assign responsibilities to team members

Clearly define roles and responsibilities for each team member to contribute fully to the development of the project [Lubelczyk and Parra, 1992]. In this way the manageable unit of work can be decreased and more tasks identified, resulting in improved supervision of individual activities [McLeod and Smith, 1996, pg 142].

4.7 Develop contingency plans to cope with staffing problems

The accurate projection of required staff levels has proven to be an absolute critical function in software development. Overstaffing may lead to higher communication and coordination overheads, which translates into lower unit productivity. On the other hand, understaffing often leads to project delays, volatile priorities and inadequate testing [Abdel-Hamid et al., 1999]. Pearlson [2001, pg 224] states that as a project progresses through its life cycle, the number of people assigned typically increases.

4.8 Include a formal and periodic risk assessment

Risk assessment consists of three activities: identifying the risks (discussed above), analysing them and assigning priorities to each of them. By analysing the risks, it is easier to understand when, why and where the risks might occur. Assigning priorities enables the project team to devote their limited resources only to the most threatening risks. For most IT projects, risk assessments are usually conducted on an infrequent and informal basis [Ropponen and Lyytinen, 1993]. According to McLeod and Smith [1996], very few organisations perform any formal risk assessment to see if risks can be controlled, reduced or managed. The management of the implementation process is often complicated by the time and effort that practitioners can devote to it. Since most organisations follow some type of software development methodology, it would be much easier to include a formal and periodic risk assessment as part of the methodology for developing systems. In this way, it is not seen as another process that has to be accomplished but instead as part of the methodology. Risks should be assessed at the beginning of the project and at the review of each phase, which should result in a decreasing exposure since they will be identified and resolved.

4.9 Divide the project into controllable portions

While most managers know that large projects must be broken down into incrementally smaller work elements, few actually dissect them to the level necessary to effectively manage them in sufficiently distinct work elements [Lubelczyk and Parra, 1992].

4.10 Educate users on the impact of changes during the project

Gioia [1996] states that as organisations implement their project plans, they often fail to view, listen and communicate effectively. To ensure that problems, issues and risks are brought to the appropriate managerial levels, the project manager should ensure that open communication is maintained with both the users and senior management [Robert, 1997].

4.11 Assess cost and schedule impact of each change to requirements and specifications

Keil et al. [1998] states, to avoid the problem of scope creep, project managers should inform users of the impact of scope changes in terms of project cost and schedule. Project managers should be able to distinguish between desirable and absolutely necessary functionality.

4.12 Stabilise requirements and specifications as early as possible

Many projects are faced with uncertainty when requirements are first stated. Requirement issues need to be resolved as the project progresses [Cule, Schmidt, Lyytinen and Kyle, 2000]. The project manager should provide the leadership and vision and make sure that the users of the system are involved in defining and stabilising its detailed requirements as early as possible [Martin et al., 1994, pg 337].

4.13 Avoid having too many new functions on software projects

McLeod and Smith [1996] stress the need to implement new functionality on an incremental basis rather than multiple new functions being incorporated simultaneously.

4.14 Review progress to date and set objectives for the next phase

Lubelczyk and Parra [1992] recommend that the project's progress should be frequently compared to the project plan. Anderson and Narasimhan [1979] point out that knowledge of the risks of failure inherent in a particular situation can enable practitioners to develop strategies (based on organisational experience and relevant theories) to reduce or overcome the influence of setbacks.

From the above, it is clear that very few control factors relate to technical issues. A majority of the issues are concerned with communications and commitments (or rather the lack thereof) among people.

5. EMPIRICAL STRATEGY

The literature study resulted in the following question – Do experienced project managers control software project risk factors with the controls identified in the literature?

This resulted in the following objectives for the empirical work:

— Identify the risks involved in a software project;
— Rank them in order of importance and frequency of occurrence; and
— Identify the activities performed by project managers to control the risks identified.

Data collection was achieved through the use of a structured questionnaire, which asked respondents questions aimed at achieving the above objectives. Ten risk factors and fourteen controls were presented to respondents. A pilot study was conducted to identify possible shortcomings of the questionnaire and to enable the questionnaire to be improved. The questionnaire was pre-tested a week prior to the actual research by emailing the questionnaire to 9 project managers. The responses resulted in improvements to the original questionnaire.

With the revised questionnaire, the method of sample selection referred to as 'snowball' sampling was used. This involved e-mailing survey questionnaires to project managers in the field and requesting names of companies or project managers that they knew that have experience in this area. This generated several more responses as these referrals were used to gain further referrals and hence the term snowball. Babbie [2001] points out that this procedure is appropriate when members of a homogeneous group (such as IT project managers) are difficult to locate. A total of 70 project managers were polled. The project managers used in this survey come from specific industries, mainly IT and Finance, even though they have managed projects for various other companies.

The analysis of returned questionnaires was conducted using a combination of rank ordering and regression techniques.

6. FINDINGS

Respondents were presented with various questions, which used Likert-type scales. For presentation purposes in this report, the more extreme categories were combined as follows, effectively reducing five-point scales to three-point scales:

For choices being headed 'completely unimportant', 'not very important', 'important', 'very important' and 'extremely important': a category called 'not that important' was created, combining the two ratings 'completely unimportant' and 'not very important'. Similarly, a category called 'very important' combined the two ratings 'very important' and 'extremely important'.

Similarly, five *frequency* categories were re-scaled into three sub-categories for presentation purposes. 'Hardly ever' combined the two ratings 'never' and 'seldom'. 'Sometimes' was unchanged, while 'most of the time', combined the two ratings 'frequently' and 'always'.

6.1 Introduction

Of the 70 questionnaires distributed, 36 were returned. All 36 questionnaires were complete and were thus used in the analysis of the data, which was done using rank order methods, frequency scales and regression analysis. The results for regression analysis indicated general trends only due to fairly low R^2 values.

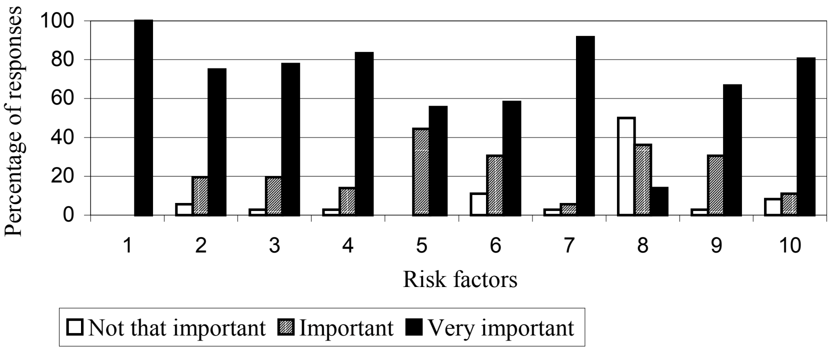
6.2 Importance of risk factors

All respondents indicated that the risk of unclear or misunderstood scope/objectives was very important. In fact, most of the risk factors were identified as very important, the only exception being 'gold plating', where only 14% of the respondents indicated that it was very important.

Aggregating the responses resulted in the following ranking of the (importance of the) listed risks: (in order of importance): (Ten of the original fourteen risks are retained)

— Unclear or misunderstood scope/objectives (Risk 1)
— Misunderstanding the requirements (Risk 7)
— Failure to gain user involvement (Risk 4)
— Lack of senior management commitment (Risk 3)
— Developing the wrong software functions (Risk 10)
— Unrealistic schedules and budgets (Risk 2)
— Continuous requirement changes (Risk 9)
— Inadequate knowledge/skills (Risk 5)
— Lack of effective project management methodology (Risk 6)
— Gold plating (Risk 8)

Table 1 shows the overall ranking of importance for each risk factor for three categories of project manager experience.



Legend:

1	Unclear or misunderstood scope/objectives	6	Lack of effective project management methodology
2	Unrealistic schedules and budgets	7	Misunderstanding the requirements
3	Lack of senior management involvement	8	Gold plating
4	Failure to gain user involvement	9	Continuous requirement changes
5	Inadequate knowledge/skills	10	Developing the wrong software functions

Figure.1. *Responses for important risk factors.*

Risk factors	Overall ranking	Experience 2-5 years	Experience 6-10 years	Experience >10 years
Unclear or misunderstood scope/objectives	1	1	1	1
Misunderstanding the requirements	2	2	2	8
Failure to gain user involvement	3	3	6	3
Lack of senior management commitment	4	4	5	4
Developing the wrong software functions	5	5	3	10
Unrealistic schedules and budgets	6	7	4	2
Continuous requirement changes	7	6	9	7
Inadequate knowledge/skills	8	8	8	5
Lack of effective project management methodology	9	9	7	6
Gold plating	10	10	10	9

Table. 1. *Importance (ranking) of risks as perceived by different levels of experience, and overall.*

Lack of senior management involvement is perceived to be less important for project managers with the most experience. This indicates that experienced project managers have overcome this risk factor and therefore do not recognise it as being that important.

The more experienced project managers place less emphasis on the importance of the risk factors, suggesting that their experience has enabled them to develop preventative and countermeasures as a matter of course.

.

6.3 Frequency of occurrence of risk factors

Respondents were also asked to indicate which of the risk factors occurred frequently. The data revealed that 'unrealistic schedules and budgets' occurs most of the time in software projects, followed by 'continuous requirement changes' and 'unclear or misunderstood scope/objectives.' The remaining risk factors do not occur as often. The frequency of occurrence of the risk factors is shown in table 2, (sequence from most to least frequent).

Risk	Risk no.	Most of the time	Some-times	Hardly ever
Unrealistic schedules and budgets	2	X		
Continuous requirement changes	9	X		
Unclear or misunderstood scope/objectives	1	X		
Lack of senior management commitment	3	X		
failure to gain user involvement	4		X	
Misunderstanding the requirements	7		X	
Gold plating	8		X	
Lack of effective project management methodology	6			X
Inadequate knowledge/skills	5			X
Developing the wrong software functions	10			X

Table. 2. *Frequency of occurrence of risk factors.*

The results are discussed in terms of the frequency of occurrence of the risks, by experience category of the project managers. The most experienced project managers stated that 'unclear or misunderstood scope/objectives' occurred most frequently. 'Continuous requirement changes' as well as 'developing the wrong software functions' are also very frequent, in the view of the most experienced project managers. These experienced managers also rate 'frequency of occurrence of lack of an effective project management methodology' as high.

The less experienced (i.e. 2 – 5 years experience) rate 'unrealistic schedules and budgets' as well as 'misunderstanding the requirements' as highly frequent. This suggests that project managers that have less experience do not have enough knowledge to control these risk factors. The perceived frequency of the risk of misunderstanding the requirements begins to decrease when more experience is gained. On the other hand, continuous requirement changes increases when more experience is gained.

Failure to gain user involvement and inadequate knowledge/skills has a distinct minority of occurrences when project managers have more than ten years of experience. Experienced project managers consider potential team members expertise before assigning them to the project team.

6.4 Frequency of occurrence of controls

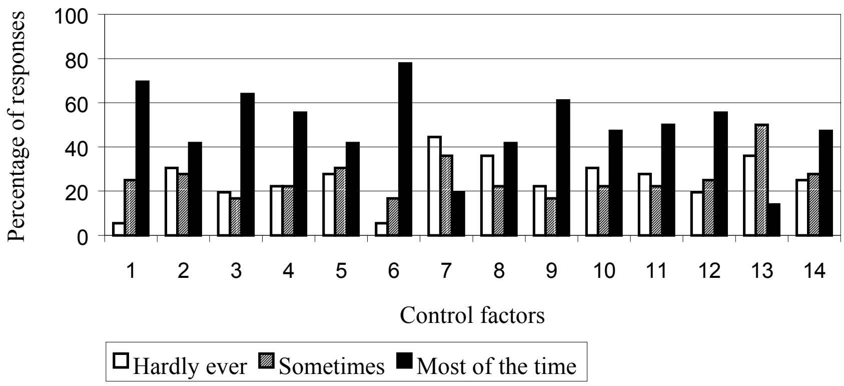
This section analyses the controls that may be performed by project managers to reduce the risks identified in the literature review.

Figure 2 illustrates the percentages of responses for each of the control factors. The results reveal that most of the controls are used most of the time. The two exceptions were control 7 (develop contingency plans to cope with staffing problems) (hardly ever) and control 13, (not attempting too many functions on software projects), (sometimes).

The most frequently used control factors in decreasing order of occurrence are:

- Clearly assign responsibilities to team members (Control 6)
- Develop and adhere to a software development plan (1)
- Management involvement during the entire project lifecycle (3)
- Dividing the project into controllable portions (9)
- Stabilise requirements and specifications as early as possible (12)
- Involving users during the entire project lifecycle (4)
- Assess cost and schedule impact of each change to requirements and specifications (11) and review progress to date (14)
- Educate users on the impact of scope changes during the project (10)
- Ensure there is a steering committee in place (5)
- Combine internal evaluations with external reviews (2)
- Include a formal and periodic risk assessment (8)
- Not attempting too many new functions on software projects (13)
- Develop contingency plans to cope with staffing problems (7)

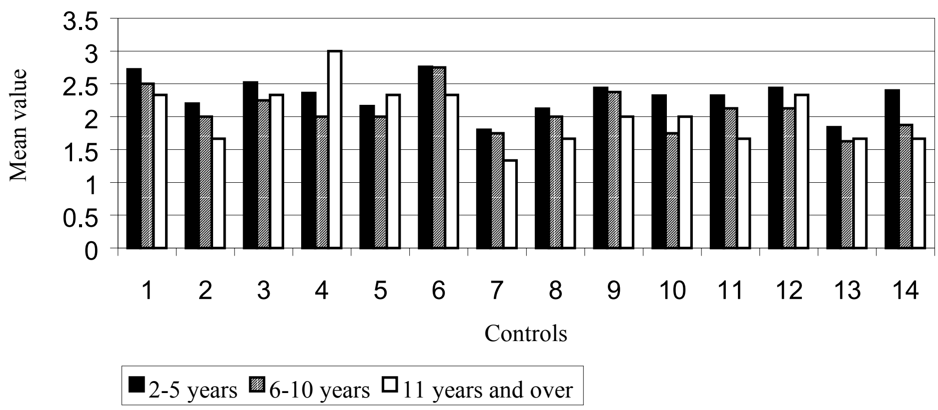
Figure 3 illustrates the frequency of occurrence of each of the controls for the three categories of experience.



Legend:

1	Develop and adhere to a software plan	8	Include a formal and periodic risk assessment
2	Combine internal evaluations with external reviews	9	Dividing the project into controllable portions
3	Management involvement	10	Educate users on the impact of scope changes
4	User involvement	11	Assess cost and schedule impact of each change
5	Ensure there is a steering committee in place	12	Stabilise requirements and specifications
6	Assign responsibilities to team members	13	Not attempting too many new functions
7	Develop contingency plans to cope with staffing problems	14	Review progress to date

Figure 2. *Frequency of occurrence of controls.*



Legend:

1	Develop and adhere to a software plan	8	Include a formal and periodic risk assessment
2	Combine internal evaluations with external reviews	9	Dividing the project into controllable portions
3	Management involvement	10	Educate users on the impact of scope changes
4	User involvement	11	Assess cost and schedule impact of each change
5	Ensure there is a steering committee in place	12	Stabilise requirements and specifications
6	Assign responsibilities to team members	13	Not attempting too many new functions
7	Develop contingency plans to cope with staffing problems	14	Review progress to date

Figure. 3. *Frequency of occurrence of controls for experience.*

Twelve of the fourteen controls are used more frequently by the less experienced (2 – 5 years) project managers, than (by) their more experienced colleagues. The remaining two controls, 'user involvement' and 'ensuring that there is a steering committee in place', are used more frequently by the most experienced project managers.

Project managers have realised that by 'developing and adhering to a software plan', the project would flow smoothly and therefore result in successful completion. However, as experience is gained a project plan is used less frequently.

'Combining internal evaluations with external reviews' is used more frequently by project managers with 2 – 5 years and 6 – 10 years experience. The more experience project managers have, the less this control is used.

'Develop contingency plans to cope with staffing problems' is seldom used by managers of any of the experience categories. As there are invariably staffing problems with IT projects, we surmise that staffing is taken into account and controlled as part of the project plan and there is therefore no need for a separate control.

'Risk assessment', which is one of the more important control factors, is not often performed. This can be concluded from all three categories of project experience, where the frequency of occurrence is very low and decreases as experience is gained. This confirms earlier literature findings.

'Dividing the project into controllable portions', 'assessing cost and schedule impact' as well as 'reviewing progress to date' also occur less frequently as experience is gained. This implies that the more experienced project managers perceive that they are very capable of controlling software projects, and have 'internalised' various controls.

6.5 Relationships between variables

Regression analysis was performed on the data to determine whether there were significant relationships between some of the demographic data (such as the project manager's number of years of experience) and the risks and controls. None of the pairings resulted in high values of R^2, so interpretation of relationships between the variables is cautious and findings are reported conservatively.

6.5.1 *Comparison of number of projects managed to frequency of risk factors*

This test was performed to determine the possibility and strength of the linear relationship between the number of projects managed, compared to how often each risk factor occurs. The findings suggest that as the number of projects managed increases, two of the risk factors ('unrealistic schedules and budgets' and 'misunderstanding the requirements' occur less frequently. (P=.06 and .07 respectively.) (No similar conclusions are possible for the other risk factors.)

6.6 Relationships between risks and control variables.

These tests were performed using regression analysis, to compare the controls to each of the risk factors to determine if they are effective in mitigating the occurrence of each risk factor. Relationships between risks and controls, which were insignificant, are not reported.

Controls	P Value
Management involvement	0.035
Dividing the project into controllable portions	0.001
Stabilise requirements and specifications	0.001

Table. 3. *Risk of 'unclear or misunderstood scope/objectives' compared to controls.*

As management involvement increases, the risk of unclear or misunderstood scope/objectives appears to decrease. The p-value shows that this relationship is significant at a 95% confidence level.

The risk of 'unclear or misunderstood scope/objectives' also occurs less frequently when the project is broken down into controllable portions. If requirements were stabilised early in the project, the risk of unclear scope and objectives would occur less frequently. These relationships are significant at a 99% confidence level.

Controls	P Value
Develop and adhere to a software plan	0.046
Combine internal evaluations with external reviews	0.003
Management involvement	0.006
Include a formal and periodic risk assessment	0.025
Dividing the project into controllable portions	0.002
Assess the cost and schedule impact of each change	0.001
Review progress to date	0.047

Table. 4. *Risk of 'unrealistic schedules and budgets' compared to controls.*

There are seven controls that have an effect on reducing the occurrence of unrealistic schedules and budgets. By performing evaluations and assessing the strengths and weaknesses of the project, this risk occurs less frequently. Management involvement has an influence over the project and they can disagree if the schedules and budgets are unrealistic, also resulting in this risk factor occurring less frequently.

Dividing the project into controllable portions ensures that the risk of unrealistic schedules and budgets is reduced. This relationship is significant at a 99% confidence level. Assessing the cost and schedule impact of each change has a high influence over the occurrence of this risk factor. By enforcing the controls in table 4, the risk of 'unrealistic schedules and budgets' is reduced.

Controls	P Value
Ensure there is a steering committee in place	0.041
Educate users on the impact of scope changes	0.046
Stabilise requirements and specifications	0.042

Table. 5. *Risk of 'failure to gain user involvement' compared to controls.*

As shown in table 5, users become more involved when there is a steering committee in place and they are educated on the impact of scope changes.

Controls	P Value
Include a formal and periodic risk assessment	0.001
Dividing the project into controllable portions	0.006
Assess the cost and schedule impact of each change	0.013
Stabilise requirements and specifications	0.000
Review progress to date	0.001

Table. 6. *Risk of 'lack of effective project management methodology' compared to controls.*

When risk assessments are performed, it is more likely that an effective project management methodology is in place. Depending on the type of methodology used, risk assessment would be included as part of the methodology and therefore the significant relationship.

Stabilising requirements and specifications also offsets the risk illustrated in table 6. This relationship is significant at a (virtually) 100% confidence level.

Controls	P Value
Assign responsibilities to team members	0.045

Table. 7. *Risk of 'gold plating' compared to controls.*

Assigning responsibilities to team members has a low impact on reducing the risk of gold plating as depicted in table 7.

Controls	P Value
Develop and adhere to a software plan	0.025
Management involvement	0.021
User involvement	0.047
Ensure that there is a steering committee in place	0.011
Include a formal and periodic risk assessment	0.004

Table. 8. *Risk of 'continuous requirement changes' compared to controls.*

There are five controls that have an impact on reducing the occurrence of continuous requirement changes as shown in table 8. Developing and adhering to a software plan helps to ensure that requirement changes are kept to a minimum. A formal and periodic risk assessment would identify the risk of continuous requirement changes.

Controls	P Value
Dividing the project into controllable portions	0.034
Stabilise requirements and specifications	0.020

Table. 9. *Risk of 'developing the wrong software functions' compared to controls.*

The risk of developing the wrong software functions is reduced to a certain extent when the project is divided into controllable portions and requirements and specifications are stabilised. The relationships between this risk factor and these controls are significant at a 95% confidence level.

6.7 Effectiveness of controls

Interpreting the relationships found in the previous section from a perspective of controls rather than risks reveals that two of the controls (each) mitigate four risks, while two other controls each mitigated three risks.

- 'Dividing the project into controllable portions' mitigates the risks of:
    - 'Unclear or misunderstood scope / objectives'
    - 'Unrealistic schedules and budgets'
    - 'Lack of a project management methodology', and
    - 'Developing the wrong functions'.
- 'Stabilise requirements and specifications as early as possible' mitigates the risks of:
    - 'Unclear or misunderstood scope / objectives'
    - 'Failure to gain user commitment'
    - 'Lack of a project management methodology', and
    - 'Developing the wrong functions'.
- 'Management involvement during the entire project lifecycle' mitigates the risks of:
    - 'Unclear or misunderstood scope / objectives'
    - 'Unrealistic schedules and budgets', and
    - 'Continuous requirement changes'.
- 'Include a formal and periodic risk assessment' mitigates the risks of:
    - 'Unrealistic schedules and budgets'
    - 'Lack of a project management methodology', and
    - 'Continuous requirement changes'.

In addition, four other controls showed similar versatility in that they each mitigated two risks.

7 CONCLUSIONS

Seven of the ten risk factors were mitigated by the use of controls, thus implying that the controls identified in the literature have proven to be very effective. Four of the controls used (each) mitigated three or more risk factors.

The most frequent risk factors, identified in this survey, that occur in software projects are continuous requirement changes, and unrealistic schedules and budgets. These two risk factors were also amongst the top six software risk factors identified by Boehm's [1991] research.

The effectiveness of each control in comparison to the risk factors was presented. The more significant results demonstrate that seven of the ten risk factors are reduced by the use of controls.

Further research directly mapping controls to risks will strengthen the above findings.

Some risk factors such as 'unrealistic schedules and budgets' and 'misunderstanding the requirements' tend to occur less frequently with greater project management experience, suggesting that methods of minimising these risks become more internalised with experience.

This research has provided a list of risks encountered in software projects, which can be used as guidelines for future software projects. The project managers used in the survey come from specific industries, mainly IT and Finance, but most of them have also managed projects for various other industry sectors.

A list of controls was also established to ensure that these risks were adequately managed and should therefore be implemented to prevent or minimise the impact of further possible problems.

8 REFERENCES

ABDEL-HAMID, T.K., SENGUPTA, K. AND RONAN, D. 1993. Software Project Control: An Experimental Investigation of Judgement with Fallible Information. *IEEE Transactions on Software Engineering,* Vol 19[6], 603-612.

ABDEL-HAMID, T.K., SENGUPTA, K. AND SWETT, C. 1999. The Impact of Goals on Software Project Management: An Experimental Investigation. *MIS Quarterly,* Vol 23[4], 531-555.

ANDERSON, J. AND NARASIMHAN, R. 1979. Assessing Implementation Risk: A Methodological Approach. *Management Science,* Vol 25[6], 512-521.

BABBIE, E. 2001, *The Practice of Social Research*. Wadsworth Publishers, USA, 9th Edition.

BOEHM, B. AND ROSS, R. 1989. Theory-W Software Project Management Principles and Examples. *IEEE Transactions on Software Engineering,* Vol 15[7], 902-916.

BOEHM, B. 1991. Software Risk Management: Principles and Practices. *IEEE Software,* Vol 8[1], 32-41.

CASH, J. JR., MCFARLAN, F.W. AND MCKENNEY, J.L. 1988. *Corporate Information Systems Management: The Issues Facing Senior Executives.* Dow Jones-Irwin, USA.

CULE, P. SCHMIDT, R. LYYTINEN, K. AND KEIL, M. 2000. Strategies for Leading off IS Project Failure. *Information Systems Management,* Spring, 65-73.

EWUSI-MENSAH, K. AND PRZASNYSKI, Z.H. 1991. On Information Systems Project Abandonment: An Exploratory Study of Organizational Practice. *MIS Quarterly,* Vol 15[1], 67-88.

GIOIA, J. 1996. Twelve Reasons Why Programs Fail. *PM Network*, November 1996, 16-20.

HEEMSTRA, F.J. AND KUSTERS, R.J. 1996. Dealing with Risk: A Practical Approach. *Journal of Information Technology,* Vol 11, 333-346.

KEIL, M. 1995. Pulling the Plug: Software Project Management and the Problem of Project Escalation. *MIS Quarterly,* Vol 19[4], 421-441.

KEIL, M., CULE, P.E., LYYTINEN, K. AND SCHMIDT, R. 1998. A Framework for Identifying Software Project Risks. *Communications of the ACM,* Vol 41[11], 77-83.

LAUER, T.W. 1996. Software Project Managers' Risk Preferences. *Journal of Information Technology,* Vol 11, 287-295.

LEHMAN, J.H. 1979. How Software Projects are Really Managed. *Datamation*, January 1979, 119-129.

LUBELCZYK, J., AND PARRA, A. 1992. *Recommended Approach to Software Development, Revision 3.* Software Engineering Laboratory, NASA, June 1992.

MAY, L.J. 1998. *Major Causes of Software Project Failures*. Available at: http://stsc.hill.af.mil/crosstalk/1998/jul/causes.asp. Accessed 25 June 2002.

MARTIN, E.W., DEHAYES, D.W., HOFFER, J.A. AND PERKINS, W.C. 1994. *Managing Information Technology: What Managers Need to Know.* 2nd Edition, Prentice Hall, New Jersey.

MCFARLAN, F.W. 1981. Portfolio Approach to Information Systems. *Harvard Business Review,* Vol 59[5], 142-150.

MCLEOD, G. AND SMITH, D. 1996. *Managing IT Projects.* Boyd and Fraser Publishing, Massachusetts.

MURSU, A., SORIYAN, H.A., AND OLUFOKUNBI, K.C. 1996. *Toward Successful ISD in Developing Countries: First Results from a Nigerian Risk Study Using the Delphi Method.* Available at: http://iris22.it.jyu.fi/iris22/pub Accessed 25 June 2002.

PEARLSON, K.E. 2001. *Managing and Using Information Systems: A Strategic Approach*. John Wiley and Sons Inc, New York.

PHAN, D., VOGEL, D. AND NUNAMAKER, J. 1998. The Search for Perfect Project Management. *Computerworld,* Vol 22, 95-100.

PHELPS, R. 1996. Risk Management and Agency Theory in IS Projects: An Exploratory Study. *Journal of Information Technology,* Vol 11, 297-307.

POWELL, P.L., AND KLEIN, J.H. 1996. Risk Management for Information Systems Development. *Journal of Information Technology,* Vol 11, 309-319.

ROBERT, D.W. 1997. Creating an Environment for Project Success. *Information Systems Management,* Winter, 73-77.

ROPPONEN, J., AND LYYTINEN, K. 1993. Can Software Risk Management Improve System Development: An Exploratory Study. *European Journal of Information Systems,* Vol 6, 41-50.

ROPPONEN, J., AND LYYTINEN, K. 2000. Components of Software Development Risk: How to Address Them? *IEEE Transactions on Software Engineering,* Vol 26[2], 98-111.