The Importance of 'Risk Radar' in Software Risk Management: A Case of a Malaysian Company

Khairul Azizan Suda
IPROM, Universiti Kuala Lumpur, Malaysia
E-mail: khairulazizan@iprom.unikl.edu.my

Nazatul Shima Abdul Rani
School of Business
Curtin University, Sarawak Campus, Miri, Malaysia
E-mail: shimarani@curtin.edu.my

Abstract: 'Risk radar' is applied to a company in Malaysia, a discussion on the implementation, implications and recommendation highlighted in this paper. The scope of this study has been an analysis of risk management and risk exposure of software projects practices in the company. This study also provided the evident that the successes of the several software that goes into the Malaysian market, depending on how risk management and its plan in software development as in the case of the selected company. It also exposed on how significant is the risk management contributing to cost effective and growth. Findings also included using 80/20 rules or Pareto Principle, 80% of the risks item listed by Boehm in Ten (10) Top Risks are due to 20% of sources (i.e. soft risks). Empirical studies have shown that 80% of the software rework comes from 20% of the problems, and that many of these critical problems involve neglect of off nominal requirements and all these negligence are caused by human (soft risk) (Boehm, 1989; Boehm-Basili, 2001; Standish Group Chaos Study Report (STANDISH), 1995). The company must implement the propose system to ensure that good practice and successful implementation of software risk management is the key factor to successful creation of software that are marketable and high quality benchmarking of plant industrial solutions. It could contribute to gain competitive advantage by at least 50% of project cost due to risks such as rework, budget overruns cost overrun, content deficiencies and etc.; ability to sustain due to minimum impact by software risks; and ability to own the technology rather than uses the technology with reasonable cost in development and always meet or exceed customer requirements.

Keywords**:** software risk management, business performance, risk radar, risk models

Introduction

According to Boehm-Basili (2001), empirical studies have shown that 80% of the software rework comes from 20% of the problems, and that many of these critical problems involve neglect of off nominal requirements. All these negligence are caused by human (soft risk). The key point, Soft risk in risk management is a main issue that contributed to rework and many other creeping problems (Boehm, 1989; Boehm-Basili, 2001; Standish Group Chaos Study Report (STANDISH), 1995). Risk is defined as a problem that could cause some loss or threaten the success of a project, but which has not happened yet (Wiegers, 1998). Other than that, Gilb (1998) defined risk as anything that can lead to results that deviate negatively from the stakeholders' 'real' requirements for a project (Gilb, 1998). In which, potential problems might have an adverse impact on the cost, schedule or technical success of the project, the quality of the products, or team morale (Boehm, 1989; Boehm-Basili, 2001;Standish Group Chaos Study Report (STANDISH, 1995). Many events occur during the course of a software development project.

Literature Review

Factors for Project Success and Failures

In a study conducted by Standish Group on IT executive managers in USA, for their opinions on why projects succeed in 1995, it showed that top three reasons for the success can be user involvement; executive management support and clear statement of requirements. The top three challenges factors identified are lack of user input, incomplete requirements and specification, and changing requirements and specification.The project-impaired factors showed that the top three factors are incomplete requirements; lack of user involvement and lack of resources. By comparison, it can be seen that project failures are more now, as compared to 10 years ago.

This has happened despite the fact that technology has had time to mature. Risks are distinguished from other project events as stated in Table 1.

Table 1: Project event risks

Project Event Risks	Details
A loss associated with the event	An event that creates a situation when something negative happens to the project. The loss associated with the risk is known as the risk impact.
The likelihood that the event will occur	A likelihood figure can be used, rather than a probability figure. The likelihood of a risk can be measured from 0 (impossible) to 5 (certainty), in the Risk Radar.
The degree to which we can change the outcome	Risk control that involves a set of actions to be taken to reduce or eliminate the risk.

Source: Standish Research Paper, Chaos Study, (1995)

Risk Database Fields

The project manager's role as integrator of project resources and information is illustrated by the task of managing diverse risk information. The use of a simple database, which by its nature manages the relationship of disparate types of data (text, dates, values), is vital to performing this task with a sense of consistency, accuracy, and convenience (Chadbourne, & Sanders, 2000).

Typical Software Risk Items

These are the risk items and risk management techniques that are typical in most software development process (refer to Table 2).

Table 2: The Top Ten Software Risk Items

Risk Item	Risk Management Techniques
Personnel Shortfalls	Staffing with top talent; key personnel agreements; incentives; team building; training; tailoring process to skill mix; peer reviews
Unrealistic schedules and budgets	Business case analysis; design to cost; incremental development; software reuse; requirements de-scoping; adding more budget and schedule
COTS; external components	Qualification testing; benchmarking; prototyping; reference checking; compatibility analysis; vendor analysis; evolution support analysis
Requirements mismatch; gold plating	Stakeholder win-win negotiation; business case analysis; mission analysis; ops-concept formulation; user surveys; prototyping; early users' manual; design/develop to cost
User interface mismatch	Prototyping; scenarios; user characterization (functionality, style, workload)
Architecture, performance, quality	Architecture trade-off analysis and review boards; simulation; benchmarking; modelling; prototyping; instrumentation; tuning
Requirement changes	High change threshold; information hiding; incremental development (defer changes to later increments)
Legacy software	Design recovery; phase-out option analysis; wrappers/mediators; restructuring
Externally performed tasks	Reference checking; pre-award audits; award-fee contracts; competitive design or prototyping; team-building
Straining computer science capabilities	Technical analysis; cost-benefit analysis; prototyping; reference checking

Source: Boehm (2001)

Risk Management Activities

Risk management involves several important steps; ***first***, the assessment of the risks of a project, by identifying what may occur during the course of development or maintenance (three activities: identifying the risks, analyzing them, and assigning priorities to each of them). ***Second,*** developed a checklist of problems that may occur; it should be reviewed to determine if the new project is likely to be subject to the risks listed if available and for a new project, the checklist should be augmented with an analysis of each of the activities in the development cycle. ***Finally***, analyze the risks that have been identified and understand as much as possible on when, why and where the problem might occur. Techniques to be used to enhance the understanding of the risks management are system dynamics models, cost models, performance models, network analysis, and more (Hall, 1998).

Models for Software Risk Management

Several risk management models had been introduced and will summarize in the Table 3 on next page. The models are Boehm's Risk **Management Process** (Boehm, 1989)**, Kontio's Risk Management Process (Riskit)** (Kontio, 2001)**, and "Soft Risk" Model** (Keshlaf, A.A. and Hashim, K. ,2000 **).**

Table 3: Summary of the models

Models	Steps/Phase
Boehm's Risk Management Process	**Phase 1:Risk assessment:** risk identification, risk analysis and risk prioritization **Phase 2: Risk control:** risk management planning, risk resolution, and risk monitoring
Kontio's Risk Management Process	**Phase 1:** Initialisation Phase **Phase 2:** Risk analysis cycle
'Soft Risk' Model	**Step 1**: Risk identification **Step 2:** Risk probability and magnitude estimation **Step 3**: Risk documentation **Step 4:** Risk assessment **Step 5:** Prioritization and highlighting the highest 10 risks **Step 6:** Monitoring (Graphic representation) **Step 7:** Controlling **Step 8:** Performing statistical operation and going back to step 1

The Roles of Risk Radar in Supporting Risk Management

A key element of risk management is maintaining a set of project risks and most important risks are prioritized from the perspective of the project team or organization. *Risk Radar* is a risk management database that is designed to help project managers identify, prioritize, and communicate project risks in a flexible and easy-to-use form. It provides standard database functions to add and delete risks, together with specialized functions for prioritizing and retiring project risks. A set of standard detailed and summary reports can be easily generated to share project risk information with all members of the development team. The number of risks in each probability/impact category by time frame can be illustrated graphically, that allows the user to visualize risk priorities and uncover increasing levels of detail on specific risks. *Risk Radar* is general and allows an impact time frame to be identified. The impact time frame is an interval over which the risk's impact might materialize. As a project draws closer to one of these time frames, it will be calculated by the program and display as the number of days to the impact time frame and its impact horizon in terms of past-, near-, mid- and far-term for each risk. A Risk Level Trend calculation is generated whenever the Risk State values (probability, cost impact, schedule impact, technical impact, and other impact) are changed. Risk Radar functions as in Table 4.

Table 4: Risk radar specialised functions

Functions	*Details*
Project Set Up	Provides functions to modify the Project information, including the Project Title, Start/End dates, and category definitions.
Risk Data-Detailed	Provides a window that allows the entry of the entire set of risk record information.
Risk Data-Summary	Provides a window that allows the entry of a subset of risk record information.
View Risk State	Provides a visual view of the complete risk level distribution (high, medium, or low) and the distribution of the risks level over the impact horizon.
Prioritize Risk	Provides functions to automatically and manually prioritize the risks in relation to each other.
Risk Reports	Provides a comprehensive set of detailed and summary risk reports that can be filtered by Risk Exposure, Risk Rank, and Risk Level.
View Retired Risks	Provides a view into all the risks that have been retired, and allows the re-incorporation of the retired risks into the active risk set.
Import Risks	Provides functions to import risks record data from other Risk Radar application databases and from previous versions (2.03 and 2.02) of Risk Radar. The functions provide a set of filter and import rules that simplifies the import process. Export Mitigation Steps-exports all the stored mitigation steps into a MS Project compatible text file that can be directly imported into a MS Project schedule.
Exit Risk Radar	Exits the application and saves any changes that were made.

Source: American Systems Corporation (ASC) *RiskRadar: The 16 Critical Software Practices* ASC White Papers. 8 March 2008

Methodology

The implementation and evaluation has been conducted prior agreement by the company management, and with anonymity of the company profile for publication purposes. The respective department are aware of the study and agreed to give cooperation for the purpose of this study. The outcome of this study has been shared with top management of the company. The duration of the study are about one year.

A Case of XXX Sdn Bhd: Using Risk Radar for Software Implementation Success of a Project

Overview of the company

XXX Sdn Bhd is one of the oil and gas companies founded in Malaysia in 1992 that is lead by its holding company XXX Group that serve oil and gas and petrochemical plant in several areas such as engineering services, local and international trading, technical training, and medical services. This study is of a high relevance since XXX Sdn Bhd has been developing several engineering software for its client especially in the oil and gas, and petrochemical industries for Asia Pacific and also Middle East region. A few years back, XXX Sdn Bhd has developed high potential engineering software as a solutions and value added to the current services. However, it exposes the project to risks such as rework, cost and budget overruns, content deficiencies, and failing to meet the deadline. It was due to qualification, knowledge, and experiences of the earlier management team who handle the software project were not from software engineering background that contributed to these risks. Risk radar is used in one of the software project development for XXX Sdn Bhd.

The Scope of the Study

The scope of this study has been an analysis of risk management and risk exposure of software projects practices in XXX Sdn Bhd. This study also provided the evident that the successes of the several software that goes into the Malaysian market, depending on how risk management and its plan in software development as in the case of XXX Sdn Bhd. It also exposed on how significant is the risk management contributing to cost effective and growth.

An overview of the significant findings of the study

The scope of this study had been an analysis of Risk Management and Risk Exposure of software projects practices in XXX Sdn. Bhd. It had covered on how the risks had been mitigated and minimized during the software development process. Human capital, process, methods and tools are the important element to ensure the success of risk management in XXX Sdn Bhd. RiskRadar had been chosen to work with Soft Risk methodology due to its close similarities. It had been selected as a tool for XXX Sdn Bhd, in managing software project risks due to the ease of use; designed by project managers; familiarity with MS Access database application; identifying, analyzing, planning, tracking, and controlling project risk; enabling proactive risk communication and decision making; independent of project size or type; and building on Institute of Electrical and Electronic Engineers (IEEE), Software Engineering Institute (SEI), and Project Management Institute (PMI) risk management principles. For this study, risks are referred to **soft or intangible risks** such as human resources, knowledge, skill sets, relationships, leadership and behaviors. **Soft or Intangible** risk management identified a new type of risk - a risk that had a 100% probability of occurring although was ignored by the organization due to a lack of identification ability. These risks directly reduce the productivity of knowledge workers, decreased cost effectiveness, profitability, service, quality, reputation, brand value, and earnings quality. Intangible risk management allows risk management to create immediate value from the identification and reduction of risks that reduced productivity. Hence, it is significant that a study on how the soft risks to be managed and implemented using improved system in XXX Sdn Bhd.

A consideration of the findings in light of existing research studies

Findings also included using 80/20 rules or Pareto Principle, 80% of the risks item listed by Boehm in Ten (10) Top Risks are due to 20% of sources (i.e. soft risks). Empirical studies have shown that 80% of the software rework comes from 20% of the problems, and that many of these critical problems involve neglect of off nominal requirements and all these negligence are caused by human (soft risk) (Boehm, 1989; Boehm-Basili, 2001; Standish Group Chaos Study Report (STANDISH), 1995). XXX Sdn Bhd must implement the propose system to ensure that good practice and successful implementation of software risk management is the key factor to successful creation of software that are marketable and high quality benchmarking of plant industrial solutions.

Recommendation for Implementation Success

Requirements for Software Implementation Success for XXX Sdn Bhd

Listed below are the requirements needed for software implementation success:

People and competencies

To ensure the implementation of Strategic Risk Management for Software Project to work successfully, Top Management of XXX Sdn Bhd should seriously look into the five (5) major scopes such as management commitment (Ishak, 2005; Porter, 1985; Hoffman and Mehra, 1999), training and education (Ishak, 2005; Porter, 1985; SPMN 2000), employee empowerment (Nakajima, 1989; Ouichi, 1981), and company policies and goals (Klusman, 1995; Porter, 1985).

Management Commitment

Change is top-down and bottom-up, the top-down approach is to provide vision and create structure; and it must be bottom-up approach to encourage participation and generate support (Moran and Avergun, 1997). The management also must provide their employees with proper tools, techniques and other facilities to allow people to synthesize the new concepts, and align them to the new way of working. XXX Sdn Bhd must have top management support, understand and committed to embark on the Total Productive Maintenance (TPM) program as it involves a cultural change that will not happen overnight. The commitment at the top must be total, both in terms of level and extent (Hoffman and Mehra, 1999). It must include the full senior management team of XXX Sdn Bhd that are fully committed to any initiative or development program. The responsibility of the top management is to actively promote motivation, ability and favourable work environment (Heap, 1992; Nakajima, 1989). The XXX Sdn Bhd's management must provide necessary training to develop a workforce of capable, motivated, and truly autonomous workers. XXX Sdn Bhd must create a favourable work environment by eliminating the psychological and physical obstacles to workers (Nakajima, 1989; Ouichi, 1981).

Training and Education

Training and education is crucial to the success of Total Productive Maintenance (TPM). Training and education is necessary to create a clear understanding of the changes required, the purpose of the change as well as the benefits to be gained ( ). Training and education also include maintenance personnel training operators to perform routine preventive maintenance tasks, operators training on the mechanics of their equipment, problem skills, and team building (Maggard and Rhyne, 1992). TPM calls for training people to improve their job skills especially training for the equipment operators. This is because one of the important goals of TPM is to raise workers' skill levels and this can be done only if there is thorough and continuous training (Nakajima, 1989; Turbide, 1995; Moore, 1997).

Employee Empowerment

Empowering the employees is one of the elements necessary to ensure success in Strategic Software Risk Management for Software Project implementation. TPM requires employee empowerment. Under TPM, production workers assume ownership of their work area and become responsible for routine maintenance of machines and equipment (Nakajima, 1989; Patterson et al., 1995). This rule should also apply to XXX Sdn Bhd as an organization Team Culture.

Company Policies and Goals

A clear, well thought-out plan supported by appropriate policies is crucial to the successful implementation of a restructured maintenance program, because a high percentage of programs fail or fall short of desired objectives due to poor planning (Klusman, 1995).

Recommendation for Software Success

The process of risk management is referring to 'The Soft Risk Model' that is designed to reduce software risks, efforts, and costs, and at the same time improve the software quality. It had been used continuously to monitor the risks until the end of the project.The 16 Critical Software Practices for Performance-based Management contain the SIXTEEN (16) practices where NINE (9) best and SEVEN (7) sustaining practices that are the key to avoiding significant problems for software development projects in XXX Sdn Bhd. These practices have been gathered from the crucible of real-world, large-scale, software development and maintenance projects (SPMN, 2000).

Adopt Continuous Program Risk Management

For XXX Sdn Bhd, risk management is a continuous process beginning with the definition of the concept and ending with system retirement. All programs need to assign a risk officer as a focal point for risk management and maintain a reserve to enable and fund risk mitigation. Risk need to be identified and managed across the life of the program. All risks identified to be analyzed, prioritized-by impact and likelihood of occurrence-and tracked through an automated risk management tool (SPMN, 2000). High-priority risks must be reported to management on a frequent and regular basis by the employees in XXX Sdn Bhd.

Estimate Cost and Schedule Empirically

Initial software estimates and schedules can be considered as high risk due to lack of definitive information available at the time they are defined. The estimates and schedules can be refined as more information becomes available. At every major program review, costs-to-complete and rescheduling can be presented to identify deviations from the original cost and schedule baselines and to anticipate the likelihood of cost and schedule risks occurring. All estimates can be validated using a cost model, a sanity check to be conducted by comparing projected resource requirements, and schedule commitments. Every task within a work breakdown structure (WBS) level needs to have an associated cost estimate and schedule and tracked using earned value (SPMN, 2000). All costs estimates and schedules to be approved prior to the start of any work and done by R&D Manager at XXX Sdn Bhd monthly.

Use Metrics to Manage

Metrics must be defined as part of definition of process, identification of risks or issues, or determination of project success factors. All metrics definition must include description, quantitative bounds, and expected areas of application. Metrics information should be used as one of the primary inputs for program decisions. The metrics program needs to be continuously monitored by the R&D Manager in XXX Sdn Bhd monthly.

Track Earned Value

Earned value project management requires a work breakdown structure, work packages, activity networks at every WBS level, accurate estimates, and implementation of a consistent and planned process.

Earned value requires each task to have both entry and exit criteria and a step to validate that these criteria have been met prior to the award of the credit.Earned value credit is binary with zero percent being given before task completion and 100 percent when completion is validated. Earned value metrics need to be collected on a frequent and regular basis consistent with the reporting cycle required at the WBS level (the lowest level of the work package, the earned value reporting should never be less than 2 weeks). Earned value is an essential indicator and should be used as an essential metric by the risk management process. It should be carried out by R&D Manager in XXX Sdn Bhd weekly.

Track Defects against Quality Targets

All programs must have pre-negotiated quality targets, which is an absolute requirement to be met prior to acceptance by the customer. Metrics need to be collected as a result of the practices used to monitor defects, which will indicate the number of defects, defect leakage, and defect removal efficiency. Quality targets need to be redefined and renegotiated as essential program conditions change or customer requirements are modified. Compliance with quality targets to be reported to customers on a frequent and regular basis, along with an identification of the risk associated with meeting these targets upon delivery. Meeting quality targets is the main subject at every program review and to be done by the R & D Manager and programmers whenever needed.

Treat People-as the Most Important Resource

The staff should be rewarded for performance against expectations and program requirements. All staff members need to be provided facilities, tools, and work areas adequate to allow efficient and productive performance of their responsibilities. The effectiveness and morale of the staff should be a factor for reward by the top management (SPMN, 2000).

Adopt Life Cycle Configuration Management (CM)

CM has two aspects: formal CM, which manages customer-approved baseline information and development CM (which manages shared information not yet approved by the customer) (SPMN, 2000). The approval for a change to controlled information must be made by the highest-level organization such as Managing Director of XXX Sdn Bhd and Oil Sdn Bhd, which last approved the information prior to placing it under CM. CM should be implemented in a centralized library supported by an automated tool and a continuous process implemented at the beginning of a program until product retirement (SPMN, 2000).

Manage and Trace Requirements

Before any design is initiated, requirements for that segment of the software need to be agreed to. Requirements tracing is a continuous process that provide the means to trace from the user requirement to the lowest level software component. Tracing shall exist not only to user requirements but also between products and the test cases. All products that are used as part of the trace need to be under configuration control. Requirements tracing should address system, hardware, and software and the process in the system engineering management plan and the software development plan (SPMN, 2000).

Use System-Based Software Design

System architecture and software design should be documented in the system engineering management plan and software development plan and audits conducted by Quality Assurance System (QAS) regularly.

Software engineering needs to participate in the definition of system architectures and should provide an acceptance gate before software requirements are defined. All architecture and design components need to be approved through an inspection prior to release to CM (R&D Department). This inspection should evaluate the process used to develop the product, the form and structure of the product, the technical integrity, and the adequacy to support future applications of the product to program needs.

Ensure Data and Database Interoperability

All data and database implementation decisions must consider interoperability issues and, as interoperability factors change, these decisions to be revisited. All data and databases should be structured in accordance with program requirements, such as the Defense Information Infrastructure (DII) Common Operating Environment (COE), in order to provide interoperability with other systems. All databases shared with the program need to be under CM control and managed through the program change process.Databases and data should be integrated across the program with data redundancy kept to a minimum, and whenever a company is using multiple Commercial off the Shelf (COTS) packages, compatibility of the data/referential integrity mechanisms to be considered in ensuring consistency between databases.

Define and Control Interfaces

Before completion of system-level requirements, a complete inventory of all external interfaces needs to be completed. All external interfaces need to be described as to source, format, structure, content, and method of support and this definition, or interface profile, to be placed under CM control. Any changes to this interface profile should require concurrence by the interface owners. Internal software interfaces to be defined as part of the design process and managed through CM. Interfaces to be inspected as part of the software inspection process. Each software or system interface to be tested individually and a test of interface support to be conducted in a stressed and anomalous test environment.

Design Twice, Code Once

All design processes should follow methods documented in the software development plan and subject to verification of characteristics, which are included as part of the design standards for the product produced that requires evaluation prior to release to CM (SPMN, 2000). This inspection should consider reuse, performance, interoperability, security, safety, reliability, and limitations. Traceability needs to be maintained through the design and verified as part of the inspection process (SPMN, 2000).

Assess Reuse Risks and Costs

The use of reuse components, COTS, Government off the Shelf (GOTS), or any other Non-Developmental Items (NDI) should be treated as a risk and managed through risk management. Application of reuse components, COTS, GOTS, or any other NDI must be made only after successful completion of a NDI acceptance inspection. This inspection needs to consider the process used to develop it, how it was documented, number of users, user experience, and compliance with essential program considerations such as safety or security. Before a decision is made to reuse a product or to acquire COTS, GOTS, or NDI, a complete cost trade-off should consider the full life cycle costs, update requirements, maintenance costs, warranty and licensing costs, and any other considerations throughout the life cycle of a product. COTS, GOTS, or NDI decisions should be based on architectural and design definitions and is traceable back to an approved user requirement. All reuse components, and COTS need to be tested individually against program requirements prior to release for testing in accordance to the program test plan. Reuse, COTS, GOTS, and NDI decisions are continuously revisited and when the as program conditions change.

Inspect Requirements and Design

All products that are placed under CM are used as a basis for subsequent development and subjected to successful completion of a formal inspection prior to its release. The inspection must follow a rigorous process defined in the software development plan and should be based on agreed-to entry and exit criteria for that specific product. All products to be placed under CM to be inspected as close to their production and conducted beginning with concept definition and ending with completion of the engineering process.

Manage Testing as a Continuous Process

All testing must follow a pre-planned process that is agreed to and funded. All tests must consider not only a nominal system condition but also address anomalous and recovery aspects of the system. Prior to delivery, the system to be tested in a stressed environment, nominally in excess of 150 percent of its rated capacities. All test products (test cases, data, tools, configuration, and criteria) to be released through CM and documented in a software version description document. Every test to be described in traceable procedures and have pass-fail criteria included.

Compile and Smoke Test Frequently

Smoke testing should qualify new capability or components only after successful regression test completion and based on a pre-approved and traceable procedure. It is run by an independent organization such as QAS and not the engineers who produced it. The test is on a frequent and regular basis which is nominally no less than twice a week, and all defects identified to be documented and subject to the program change control process that is visible to all project personnel (SPMN, 2000). The implementation plan is crucial in ensuring the success of software risk management to ensure the smoothness of software development in XXX Sdn. Bhd.

Contribution of the study

If Risk Radar is implemented properly, it could contribute to XXX Sdn Bhd as the following:-

- Gaining more competitive advantage by recovering at least 50% of project cost due to risks such as rework, budget overruns cost overrun, content deficiencies and etc.
- Ability to sustain compare to close competitors that offer the similar products due to minimum impact by software risks
- Ability to own the technology rather than uses the technology with reasonable cost in development and always meet or exceed customer requirements

Limitations

The authors had assumed that the analysis done in this project is based on the assumptions that the current political, economical, social and technological (PEST) changes remained the same. The authors also had assumed that XXX SDN BHD had maintained all its current software research and development activities, engineering services and all its business activities and partnership within and after the time frame of this project. Furthermore, it is not main objective of this study to expose the financial details that contributed by the implementation of the strategic software risk management.

Implications

Implications for this study will be on the requirement issues and management issues. Most projects face uncertainty and turmoil around the product's requirements. While some of the uncertainty is tolerable in the early stages, the threat to success increases if such issues are not resolved as the project progress. Followings are possible risk factors such as lack of a clear product vision, lack of agreement on product requirements, inadequate customer involvement in the requirements process, un-prioritized requirements, new market with uncertain needs, rapidly changing requirements; ineffective requirements change management process, and inadequate impact analysis of requirements changes. Management shortcomings inhibit the success of many projects; the project manager is usually the person who is writing the risk management plan, and most people unwilling to air their own weaknesses in public. These are few of the management issues arises during the risk management plan preparation that include inadequate planning and task identification, inadequate visibility into actual project status, unclear project ownership and decision making, unrealistic commitments made, sometimes for the wrong reasons, managers or customers with unrealistic expectations and staff personality conflicts.

Recommendations

Forwarded few recommendations for companies, managers and future research for software risk management. For a business organization and any companies to be more efficient and productive, a software risk management should be installed in the organization. Without the proper system, it will result lost of data, information and business deals.

Other than that, it will reflect towards the inefficiency of the business operation for failing to observe the need to have a proper precaution system installed especially in monitoring the software functionality in supporting the business operations. Managers should be trained and be aware of the need to have a proper system installed in order to ensure the performance of a company is up to higher standard as compared to competitors. In the future the cost analysis should be included in the analysis, although it is a confidential record for the XXX Sdn Bhd, however, to really forecast the result of the failure to manage the soft risk or human factor that hinder the success of the software development should be measured by the total of losses incurred due to mismanagement of human factors.

Conclusion

There is one final aspect to be considered in any degree of project failure, all success is rooted in either luck or failure. If we begin with luck, then we will learn nothing but arrogance. However, failure begets knowledge, as such, if we begin with failure and learn to evaluate it, than we will learn to succeed. This study had served its purpose in outlaying the importance aspects of human management in ensuring the success of software development in any organization. Not only that, the success of software implementation also highly dependent on human acceptance of the new technology or software (Ishak, 2005; Porter, 1985; Hoffman and Mehra, 1999; Nakajima, 1989; Ouichi, 1981 Klusman, 1995; SPMN 2000). Human resistance in accepting the new technology or software will indirectly determine the failure or success of any organization in this information age era (Pohl , 2004; Hehn, 1999; Michael,1995).

References

Aguilar, F. J. (1967), *Scanning the business environment*. New York, NY: Macmillan Co.

American Systems Corporation (ASC) (2008). *RiskRadar: The 16 Critical Software Practices* ASC White Papers. 8 March 2008 http://www.americansystems.com/NR/rdonlyres/F8A5BD1B-681F-4899-AE59-E173D69EE09A/0/RiskRadar_16CriticalSoftwarePractices.pdf

Boehm, B. W. (1989). *Tutorial: Software Risk Management*, Les Alamitos, CA, IEEE Computer Society.

Boehm. B. W. (1991). *Software risk management: principles and practices.* IEEE software vol 8, no 1, pp 32-42.

Boehm, B. and V. Basili (2001)., *Software Defect Reduction Top 10 List*, IEEE *Computer*, January.

Dorofee , A. J. , , J. Walker, A., Albert, C. J., Higuera, R. Pl, Murphy, R. L., & Williams, R. C.(1996),

Gilb, T. (1998). *Risk Management Technology : A practical toolkit for identifying,analyzing and coping with project risks*. Version 3.0 Sept.

Hafner, A. W. (2001). *Pareto's Principle: The 80-20 Rule*. March 31, 2001. http://www.bsu.edu/libraries/ahafner/awh-th-math-pareto.html

Hall E. (1998), Managing Risk: Methods for Software Systems Development. Reading, MA: Addison Wesley, pp. 2.

Heap, J. (1992). *Productivity Management: A Fresh Approach*, London: Cassell Educational PLC,.

Hehn, Herman F.(1999). *Peopleware*: Como Trabalhar o Fator Humano nas Implementações de Sistemas Integrados de Informação (ERP). São Paulo: Ed. Gente.

Hoffman, J., & Mehra, S., Operationalizing productivity improvement programs through total quality management, 1999 *International Journal of Quality & Reliability*, 16(1), pp. 72-84.

Integrated Computer Engineering. *A Directory of American System: Testimonials*, 2008. http://www.iceincusa.com/Company.aspx?p=Company_Testimonials

IEEE, Standard for Software Life Cycle Processes – Risk Management; Std. 1540-2001

Ishak, Z.M. (2005). Total Productive Maintenance Course Module, Warwick Manufacturing Group, University of Warwick.

Keshlaf, A.A. and Hashim, K. *A Model and Prototype Tool to Manage Software Risks*, Proceedings of the First Asia-Pacific Conference on Quality Software, 2000, pp.297-305.

Kontio, J. (2002). *Risk Management Training,* University of Oulu, publisher: R&D Ware

Maggard, B. N., & Rhyne, D. M. (1992), Total productive maintenance: A timely integration productivity, *Production and Inventory Management Journal*, 33(4), 6-9.

Michael, G. de. (1995). *Computer Support for Cooperative Work: Computers between Users and Social Complexity*. Milan: COMIC Esprit Basic Research Project 6255, 1995.

Porter, M.E. , (1985). *Competitive Advantage*, Free Press, New York.

Moore, R. (1997). Combining TPM and reliability-focused maintenance, *Plant Engineering*, 51(6), 88-90.

Moran, J. & Avergun, A. (1997). Creating Lasting Change, the *TQM Magazine*, 9(2), 146-151.

Nakajima, S. *TPM Development Program*, 1989. Oregon: Productivity Press.

NASA Glenn Research Center, *Software Risk Management database* http://tkurtz.grc.nasa.gov/risk/

Patterson, W. J., Fredendall, L. D., Kennedy, W. J., & McGee, A. *Adapting total productive maintenance to Asten, Inc.*, 1996. *Production and Inventory Management Journal*, 37(4), 32-37.

PMBOK, (2000). *A Guided to the Project Management Body of Knowledge (PMBOK Guide),* 2000 Edition

Pohl J., (2004). *Interoperability and the Need for Intelligent Software*; 6th Office of Naval Research (ONR) Workshop on Collaborative Decision-Support Systems, 2004, Quantico, VA, Sep.8-9.

Standish Research Paper, *Chaos Study*, (1995), available on-line at http://www.standishgroup.com

STRA *Software Technical Risk Advisor*, Centre of Software Engineering, University of Southern California, Computer Science Department, http://sunset.usc.edu/available_tools

Software Program Manager's Network (SPMN), *16 Critical Software Practices for Performance-Based Management*, 1996-2000, Integrated Computer Engineering, Inc.

Turbide, D. A. (1995). Japan's new advantage: Total Productive Maintenance, *Quality Progress*, 28(3), 121-123.

Toth L., (2007). American System's Perspective Journal, *Enterprise IT Risk Management: A Success Story*, July 2007. http://www.americansystems.com/NR/rdonlyres/21CB8891-920D-4C58-9E38-F9FCBF87B7A3/0/Perspective_July2007.pdf