Article can be accessed online at http://www.publishingindia.com

Perceptions on Risk Management Strategies in Software Development

Nitin Deepak*, Shishir Kumar**

Abstract

According to CHAOS 2004, maximum projects were finished with overtime and overbudget, which cause project failure. Therefore this risk should be analyzed to understand software risk. Many authors have identified different risk factors like team risk, organisational and environmental risk, requirements risks, plan and control risk, etc. Solutions are not many, but simulation and case studies are some of the solutions to reduce risk. In this paper, open source architecture is discussed and also a solution is proposed, which makes the development more easy and secure towards risk.

Risk and Risk Management

CHAOS quoted that the lucrative completion of software projects was 29%, 18% were aborted projects and 53% were completed fruitfully but with overtime and over-budget. The notified reasons of such failure were the frequent changes/advancement in technology, flood of industry requirements or de-consideration of risk factors (Hartmann, 2006).

Software development is of course wealthy and prosperous in planning a unique chance, but it always contained elevated altitude of uncertainty. That's why the importance of managing the uncertainty to recover the irrecoverable development costs to some extent. Risk is not bad in itself rather it is crucial to software development and the study of assessment and analysis is truly significant.

Risk

It can be defined as all the possibilities of suffering losses. When one is dealing with software development project, quality degradations, augmented development expenditures, overtime completions or failures are the reasons of facing loss.

Barry Bohem introduces risk in the spiral process model first to consider the risk aspect in SDLC (System Development Life Cycle)

Van Scoy (Roger, 1992) has mentioned that technical risk lies in the soul of many problems causing the failure of software programs. They had defined the technical risk as "the use of theory, principles and techniques contained in software engineering aspects will fall short for capitulating the right software product. Basic technological factors that may be the reason for the final product to be very expensive, late delivery, or not accepted to the client or customer."

Project Management Body of Knowledge (Miler & Górski, 2002a), defines risk as "Tentative occurrence or stipulation that has an optimistic or a pessimistic effect on a project idea." In fact, risk management is actually a project contained by an original project.

Risk Management

Continuous Risk Management (CRM) was emphasized aiming prior identification and recognition of risks and then to take appropriate actions to the edge of and to minimize the risk (Miler &Górski, 2002b).

* Ph. D. Scholar, Bhagwant University, Ajmer, Rajasthan, India. Email: nitin.d12@gmail.com
** Professor, Jaypee University of Engineering & Technology, Guna, Madhya Pradesh, India. Email:dr.shishir@yahoo.com

Traditional and risk-oriented are the two approaches to software risk management. As the problem occurs, the generic solution which can be applied to almost all and that is reactive in nature dealing with software projects systematically, is termed as traditional approaches. The other approach is proactive in nature, as it seeks early identification and management uniquely for specific projects (Johnson, 2009).

Understanding Software Risk

Failing to understand and manage software project risk can result to a variety of problems, including cost and schedule overruns, non-meeting of user requirements, and the production of systems that are not used or don not deliver business value. As organisations continue to invest time and resources in strategically important software projects, managing the risk associated with them becomes a critical area of concern (Kumar, 2002, Osmundson et al., 2003).

When managers deal with risk, they seek to influence their environment so as to reduce negative outcomes (MacCrimmon & Wehrung, 1984; March & Shapira, 1987). Although several risk factors have been published (e.g. Alter & Ginzberg, 1978; Heemstra & Kusters, 1996; Moynihan, 1997; Schmidt et al., 2001), until relatively recently there has been little attempt to move beyond checklists and frameworks.

Author identifies six dimensions of risk (Wallacea, Keilb, & Raic, 2004):

The Table 1 summarizes all six dimensions detailed below:

Team Risk: Insufficient knowledge, lack of cooperation, lack of motivation, communication issues among team members cause uncertainty of achieving project's completion.

Organisational Environmental Risk: Unstable decision from management, organisational politics, no monitoring or cross-checking on deadlines, etc. cause poor performance of the project causes uncertainty of project completion.

Requirements Risk: Incorrect, unclear, ambiguous, inadequate or unusable requirements play a major role in project failure.

Planning & Control Risk: Unrealistic schedules and budgets, excessive pressure to complete tasks with deadlines, lack of visible milestones are some points which occur because of poor planning & control which makes it a high risk factor.

User Risk: Lack of user involvement or lack of cooperation from the user can be called as the major risk to software project completion.

Complexity Risk: Use of technology or new technology never used before, largest project attempt by the organisation first time, a large number of external links to other systems required its leads to complex structure which again tend towards high risk.

An enormous amount of decision making ability is required to deal or handle with people oriented and technological driven success factors involved in a software project.

Table 1: Risk Factors

Risk Factor	*Essence*
Team Risk	Uncertainty of task accomplishment due to lack of knowledge, motivation and communicative issues.
Organisational Environmental Risk	Uncertainty of task accomplishment due to poor management level decisions, no monitoring on deadlines etc.
Requirements Risk	Incorrect, unclear, ambiguous, unrealistic requirement cause failure of projects.
Planning & Control Risk	Unrealistic schedules and budgets, excessive pressure of deadlines, lack task visibility because of poor planning and control.
User Risk	Lack of cooperation and involvement of user causes failure of projects.
Complexity Risk	Larger projects attempt first time, use of new technology, and excessive use of external links etc. leads to complexity structure.

Risk Assessment on Distributed Projects

Changeability in domain knowledge and communication between the teams are important factors to consider in schedule risk assessment models of world-wide distributed software development projects. Different researchers in the field of global software engineering (Ebert, Murthy & Jha, 2008; Hillegersberg & Herrera, 2007; Hussey & Hall, 2008; Prikladnicki & Yamaguti, 2004) have identified common aspects that can affect project schedule, such

as: cultural differences, high number of distributed sites; different knowledge expertise and domains; many communication dependencies; time zone differences.

Adailton Lima (2010) proposes simulation-based approach to derive the project structure and statistics from a project management database to generate a stochastic simulation model. In the model proposed by him, each site has its own domain knowledge, capabilities and a set of features that have been allocated to them. To authenticate, author needs a plan to apply a case study to evaluate the accuracy of the simulation model and a survey targeted to project managers to evaluate the applicability of the tool on an industrial context.

The simulation model was used to derive a plot of project schedule risk as a function of communication probability.

Management of Risk of Web-Distributed Software Development Projects

The web application is the most popular example of developments in the web. A web application is defined by Mendes as "an application delivered over the Web that combines characteristics of both Web Hypermedia and Web Software application"(Mendes & Mosley, 2006).

The importance of web-risks is different from others in a number of ways:

- Estimation of risk probability and loss is much more difficult because of the involved challenges and relative lack of experience with them (Mendes & Mosley, 2006).
- The security threats are comparatively higher in the web applications (Glisson & Welland, 2005; Romero, Villegas & Meza, 2008; Huanget al., 2004; Geet al., 2006).
- Additional risk sources related to the W-D environment include communication, culture, diversity and difference in geographical locations (CA, 2008; Kappelet al., 2006; Bruno, Tam & Thom, 2005; Pressonet al., 2009).

Ideally, assessment and management of web application development risks should be analyzed during the complete life cycle of the projects (Keshlaf & Riddle, 2010), but unfortunately, most of the web developers use a reactive risk strategy. This approach is unsatisfactory as it composes software projects exposed to any nature of risks at any time exclusive of efficient assessment and control (Keshlaf & Riddle, 2010).

There is no other way to avoid risk for web development rather to manage them is the final solution.

Risk Management and Risk Assessment

Framework for SRM by SEI is supported by three groups:

1. Software Risk Evaluation (SRE)
2. Continuous Risk Management (CRM)
3. Team Risk Management (TRM)

There are two approaches to Risk assessment.

Reactive: Deals with problems generic to all software projects systemically and project specific problems as they arise.

Proactive: Identifies and manages unique aspects of a specific project before they impact the project.

Reactive and Proactive approaches can definitely help in identifying risk in web-distributed software projects.

Some existing tools and models of Risk Management reviewed are as mentioned below-

DS-RM Concept: Distributed Software - Risk Management Concept has been designed on the basis of idea that communication and continuous risk assessment plays a vital role in managing the risks. Three concepts are followed in this approach and have been reviewed for risk identification, snapshots for analysis and reports for assessment (Keshlaf& Riddle, 2010).

EBIOS Methodology: This method has been introduced by Central Directorate of Security of Information Systems (DCSSI) in the French government. It consist of five phases: Context study, Security Requirements Checklist, Threats study, Identification of Security Objectives and Determination of Security Requirements(Romero, Villegas, & Meza, 2008; ENISA, 2009). EBIOS could be used for security risk management in W-D (Keshlaf & Riddle, 2010).

PRO-RISK Framework: It is an open system where every user can develop, adjust anoption of the published templates or use different templates to accommodate their

project requirements. It is a framework for small and large software projects for the management of risk(Keshlaf& Riddle, 2010).

Riskit Process: Proposed in 1996 by Professor Jyrki Kontio, it is a widespread risk management technique based on theoretical principles with a wide-ranging process definition that chains risk management activities. Somehow it can also be used in W-D (Kontio, 1997).

CMMI-RSKM: This is a Risk Management process area and it has been accepted and applied universally by numerous organisations (Williams, 2006).

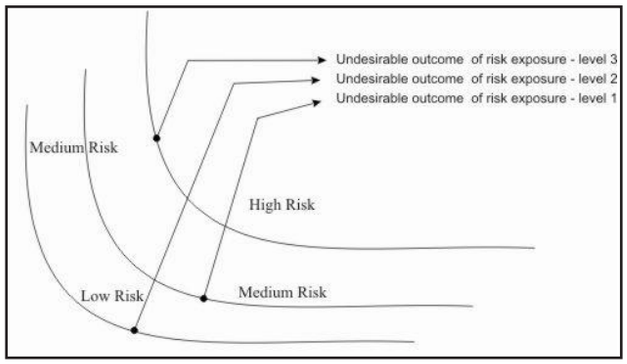
PMBOK RM Process: Project Management Body of Knowledge is a process containing four phases: Initiating, Planning, Executing and Closing (Callegari & Bastos, 2007).

GDSP RM Framework: Geographically Distributed Software Project (GDSP) is an integrated framework to handle risks in distributed software projects. It emphasizes on many aspects which are shared between GDSPs and web application developments (Pressonet al., 2009).

After analysis, the comparison has been conducted on available literatures on criteria factors (risk areas and characteristics) shows that there are only a few factors supported by the industry persons that can be used for risk management for W-D software projects (Keshlaf & Riddle, 2010).

IT Outsourcing Risk exposure

Figure 1: Probability of Undesirable Outcome i (Uo3)



A study of managerial perspectives on risk and dealing with risk hypothesize that managers do not compare the possible outcomes with the risk of alternatives and also they do not take care of indecision about positive outcomes as a feature of risk. The magnetism of an alternative comprises of strong positive outcomes, where the negative outcome is related to risk. It is also a fact that the loss extent due to negative outcome is significant (March & Shapira, 1987).

Unforeseen change over and management costs, switching costs, expensive agreement revisions, clashes and court cases, service dishonour, increase in cost, loss of organisational competency, hidden service costs etc. are some adverse outcomes from managerial facets.

Perception of Reducing Risk is to Avail Some Principles for Secure Software Development

The security aspect of software development is generally ignored by maximum developers, which leads to many security weaknesses (Jurjens, 2002). Jurjensargues that software developers mostly rely on their intuition in developing secure software, and do not use much systematic help for guidance.

So to reduce risk and considering security as a software/ web application/apps developer one needs to follow some guidelines for secure software/web application/apps development.

Many organisations are the sources of software development and information security standards like:

- Institute of Electrical & Electronics Engineers (IEEE)
- International Standards Organization (ISO)
- American National Standards Institute (ANSI)
- American Department of Defense (DoD)
- The British Standards Institute (BSI) and
- The Object Management Group (OMG)

Security Standards and Best Practices

(ISO, 1998, 2000, 2004, 2005)

ISO/IEC 27002

ISO/IEC TR 13335

NIST Security Models

Interconnection Standards

Software Development Standards and Practices

ISO/IEC 12207 defines a framework for software life cycle process. Similarly, three principal processes are identified in the IEEE 1074 standard for software development namely (Futcher & von Solms, 2008):

- Requirements
- Design
- Implementation

However, six best practices are significant importance to software development in general (Futcher & von Solms, 2008):

- Develop software iteratively
- Manage Requirements
- Use Component-based architectures
- Visually model software
- Verify software quality
- Control changes to software

Guidelines for Secure Software Development

Guidelines are categorized mainly into three areas:

Elements in the software development process

- Define security requirements
- Perform threat modeling
- Applying coding and testing standards
- Conduct code reviews

Managing the software development process

- Integrate security into SDLS
- Define security roles
- Provide security education and training
- Perform Risk Management

Security functions to be built into applications

- Determine relevant security services
- Implement appropriate security controls and mechanism

Table 2: Perception of the Risks Identified

Perception	*Detail*	*Solution*
Six dimensions	Team Risk, Organisational and Environmental Risk, Requirements Risk, Planning & Control Risk, User Risk, and Complexity Risk	Decision making power is required in dealing these six dimensions
Distributed projects	Software Distributed Web distributed	Simulation Model
Development outsourcing	Uncontrollable situation Untested components	Situation oriented solutions might be based on some previous case studies

Suggestive Aspect to Reduce Risk

As Risk was analyzed by many authors,the researcher would like to suggest reducing risks on specific risk factors like:

- Requirement changes risk
- User risk
- Team risk
- Use of outsourced components risk

Nowadays no language or model supersedes to one another. Many languages or matching modelsare equally placed in the market in the modern era. The most important aspect to be noticed about today's technology during development is that clientage is mostly online. Many software industries are moving via online clients in development. As one-to-one-clientage are almost obsolete in this modern era.

In fact, developers and client are both available on the web these days 24x7. Clients post their projects on the web and abided by developers to develop at the client satisfaction.

Due to the written and documented requirement online, requirement never changes after proposal accepted with deadlines and contract. So, risk of changing requirement decreases. If requirement changes, it means new module-new development-new contract.

During software/web applications/apps development the problem arises of maintaining customizable software, themes and components for re-usability when

outsourcing, so that it could be attached easily with other similar package(s).

Working in teams is another problematic aspect these days with full mobility and on web distributed projects, so to make the project successful.

Proposed open-source Model-View-Controller (MVC) model (www.codeigniter.com) in web/mobile development is the finest solution where team can put effort to follow the standards instructed and developed the application successfully and also as a good team-mate in the project.

Model-View-Controller (MVC) is an object oriented based approach using language PHP/ .Net/ JAVA, as the name suggests it has three different phases of development viz. Layout designing and validations, business logic, and data storage modeling.

In this MVC approach developers can change view/ layout for different stages viz. Web browsers or mobile browsers without distributing business logic and data modeling to shift accordingly, it obviously needs some good experience.

Figure 1.1: MVC Architecture

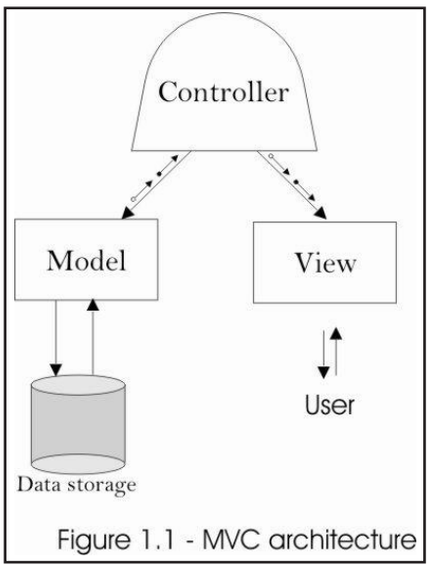


Figure 1.1 - MVC architecture

Model-View-Controller (MVC)

(www.codeigniter.com; www.cakephp.org)

Fig.2 shows

The model

It corresponds to the data organisation. Usually your model classes will include methods that help you fetch, insert, and update information in your database.

The view

It is the information that can be accessible to the user. A View in generalis a web page, but in CodeIgniter, (an MVC framework) a view may be a fragmented web pageresembling header or footer. It could also be an RSS page, or any other type of "page" which is viewable.

Controller

It provides as amediator between the Model, the View, and any other resources needed to process the HTTP request to generate a web page.

The Objective of Developing MVC Approach

Dynamic Instantiation: In Code Igniter (MVC framework), classes are loaded and routine run only when requested, rather than publicly. No hypothesizes are made by the system regarding what possibly will be required beyond the nominal core resources, so the system is light-weight by default. The events, as produced by the HTTP request, and the controllers and views designed will determine what are invoked.

Loose Coupling: The extent which components of a system depend on each other. When fewer components rely on each other the more reusable and flexible the system turns out to be. The purpose was to develop a very loosely coupled system.

Cohesiveness within components: Cohesion is the degree to which components have anintended strength or closely focused purpose. In Code Igniter (MVC framework), each class and its functions are highly independent in order to allow maximum utility/ reusability.

Merits of MVC

Using an MVC approach, an organisation can enjoy the advantages as mentioned below:

- The developer can make their own function to be reused by their encoded names, which enhance security measures.
- The team can be led efficiently by making them separate for Model, View and Controller.

- Already longed method(s) are now cooked in MVCs, which makes development fast at every level.
- Each path, help and library, etc are well documented, which surely help in reducing risk of non-understandable code which makes finding errors easier in maintenance
- Working in teams on a single project is superbly easy without affecting other team-member's work/role.

Conclusion

Different perceptions for risk management strategies in software development has discussed in this paper. Risk can be categorized as risk in team, risk in organisation and its environment, risk in requirement, planning &control, etc. Many different existing tools to manage risk were also reviewed. Two different strategies like simulation and case studies were also presented as a solution to reduce risk. This paper also suggests the use of open source architecture named M.V.C. (Model View Controller) model may conceptually help in minimizes some risk factors. This architecture can further be framed practically in team and development in future.

References

Alter, S. & Ginzberg, M. (1978). Managing uncertainty in MIS implementation. *Sloan Management Review*, 20(1), 23-31.

Bruno, V., Tam, A. & Thom, J. (2005). *Characteristics of Web Applications that Affect Usability: A Review*. In Proceedings of the 17th Australia conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future (OZCHI 05), (vol. 122, pp. 1-4). Canberra, Australia.

CA/Wily. (2008). White Paper: Application Performance Management, Effectively Managing High-Performing Business- Critical Web Application. Business-Critical Web Applications.

Callegari, D. & Bastos, R. (2007). *Project Management and Software Development Processes: Integrating RUP and PMBOK*. In Proceedings of the 2007 International Conference on Systems Engineering and Modeling. (pp. 1- 8).

Chowdhury, A. A. M. & Arefeen, S. (2011). *Software Risk Management: Importance and Practices*. 2(1), 49-54.

Curphey, M. (2004). Software Security Testing: Let's Get Back to Basics. Retrieved from SoftwareMAG.com

Ebert, C., Murthy, B. K. & Jha, N. N. (2008). *Managing Risks in Global Software Engineering: Principles and Practices*. 2008 IEEE International Conference on Global Software Engineering. (pp.131-140)

ENISA (2009).Ebios Product Identity Card. ENISA Retrieved from http://www.enisa.europa.eu/rmra/methods_tools/m_ebios.html (accessed on June 17, 2009).

Erdogmus H. (2002). Aligning Software Development Investment Decisions with the Markets. National Research Council of Canada.

Futcher, L. & von Solms, R. (2008). Guidelines for Secure Software Development.

Ge, X., Paige, R., Polack, F., Chivers, H. & Brooke, P. (2006). *Agile Development of Secure Web Applications*. In Proceedings of the 6th International Conference on Web Engineering (ICWE'06) (pp. 305-312).

Glisson, W. & Welland, R. (2005). *Web Development Evolution: The Assimilation of Web Engineering Security*. In Proceedings of the Third Latin American Web Congress (LA-WEB'05), IEEE Computer Scociety. (pp. 5).

Hartmann, D. (2006). Interview: Jim Johnson of the Standish Group. Retrieved from http://www.infoq.com/articles/Interview-Johnson-Standish-CHAOS (accessed on March 8, 2009).

Heemstra, F. J. & Kusters, R. J. (1996). Dealing with risk: A practical approach. *Journal of Information Technology*, 11(4), 333-346.

Hillegersberg, J. V. & Herrera, M. (2007). *Tool Support for Distributed Software Development: The past - present - and future of gaps between user requirements and tool functionalities*. In Tools for Managing Globally Distributed Software Development, Munich, Germany, 2007.

Huang, Y., Tsai, C., Lee, D. & Kuo, S. (2004). *Non-Detrimental Web Application Security Scanning*. In Proceedings of the 15th International Symposium on Software Reliability Engineering (ISSRE'04): IEEE Computer Society. (pp. 219-239).

Hussey, J. M. & Hall, S. E. (2008). *Managing Global Development Risk*. USA, Florida: Auerbach Publications.

ISO. (2005). ISO/IEC 27002: Information Technology - Code of Practice for Information Security Management.

ISO. (2004). ISO/IEC 13335-1: Information Technology- Security Techniques- Management of Information and Communications Technology Security. Part 1: Concepts and models for information and communications technology security management.

ISO. (1998). ISO/IEC TR 13335-3: Information Technology – Guidelines for the Management of IT Security. Part 3 : Techniques for the management of IT security.

ISO. (2000). ISO/IEC TR 13335-4: Information Technology – Guidelines for the Management of IT Security. Part 4: Selection of safeguards.

Johnson, D. L. (2009). Risk Management and the Small Software Project. Retrieved from http://www.sei.cmu.edu/iprc/sepg2006/johnson.pdf (accessed on May 4, 2009)

Jurjens, J. (2002). Using UMLSec and Goal Trees for Secure Systems Development. *Communications of the ACM*, 48(5), 1026-1030.

Kappel, G., Proll, B., Reich, S. & Retschitzegger, W. (2006). *Web Engineering the Discipline of Systematic Development of Web Application*. John Wiley & Sons, Ltd.

Keshlaf, A. A. & Riddle, S. (2010). *Risk Management for Web and Distributed Software Development Projects*. The 5th International Conference on Internet Monitoring and Protection.

Kontio, J. (1997). *The Riskit Method for Software Risk Management*. Maryland: University of Maryland.

Kumar, R. (2002). Managing risks in IT projects: An options perspective. *Information and Management*, 40(1), 63-74.

Lanowitz, T. (2005). Now is the Time for Security at the Application Level.

Lima, A. M. (2010). *Risk Assessment on Distributed Software Projects*. Cape Town, South Africa.

MacCrimmon, K. R. & Wehrung, D. A. (1984). The risk in-basket. *Journal of Business*, 57(3), 67-387.

March, J. G. & Shapira, Z. (1987). Managerial perspectives on risk and risk taking. *Management Science*, 33 (11), 1404-1418.

Mendes, E. & Mosley, N. (2006). *Web Engineering*. Berlin Heidelberg: Springer-Verlag.

Microsoft Security Intelligence Report (2008). Based on data from the DHS NVD & CERT

Miler, J. & Górski, J. (2002a). *Towards an Integrated Environment for Risk Management in Distributed Software Projects*. 7th European Conference on Software Quality, Finland.

Miler J. & Górski J. (2002b). *Supporting Team Risk Management in Software Procurement and Development Projects*. Proceedings of 4th National Conference on Software Engineering, Poland, 2002

Moynihan, T. (1997). *How Experienced Project Managers Assess Risk*. IEEE Software, 14(3), 35-41.

Osmundson, J. S., Michael, J. B., Machniak, M. J. & Gross, M. A. (2003). Quality management metrics for software development. *Information and Management*, 40(8), 799-812.

Presson, J., Mathiassen, L., Jesper, B., Madsen, T. &Steinson, F. (2009). *Managing Risks in Distributed Software Projects*: An Integrative Framework.IEEE Transaction on Software Engineering. (vol. 56, pp. 1-25).

Prikladnicki, R. & Yamaguti, M. H. (2004). *Risk Management in Global Software Development: A Position Paper*. In Third International Workshop on Global Software Development (GSD 2004), Edinburgh, Scotland, UK.

Roger, V. S. L. (1992). *Software Development Risk: Opportunity, Not Problem*. SEI, CMU/SEI-92-TR-30, ADA 258743.

Romero, B., Villegas, M. & Meza, M. (2008). *Simon's Intelligence Phase for Security Risk Assessment in Web Applications*. In Proceedings of the Fifth International Conference on Information Technology, IEEE Computer Society. (pp. 622-627).

Schmidt, R., Lyytinen, K., Keil, M. & Cule, P. (2001). Identifying software project risks: An international Delphi study. *Journal of Management Information Systems*, 17(4), 5-36.

Wallacea, L., Keilb, M. &Raic, A. (2004). Understanding software project risk: A cluster analysis. *Information & Management*, 42(1), 115-125.

Williams, R. (2006). *The CMMI RSKM Process Area as a Risk Management Standard*. In Proceedings of Sixteenth Annual International Symposium of the International Council On Systems Engineering (INCOSE).

www.codeigniter.com, MVC framework

www.cakephp.org, MVC framework