Web and Distributed Software Development Risks Management: *WeDRisk* Approach

Ayad Ali Keshlaf
School of Computing Science
Newcastle University
Newcastle, UK
a.a.a.keshlaf@ncl.ac.uk
ayadali2005@gmail.com

Steve Riddle
School of Computing Science
Newcastle University
Newcastle, UK
steve.riddle@ncl.ac.uk

Abstract—**In spite of a variety of software risk management approaches, the software industry is still suffering from associated risks. Web and distributed software development is an example, where there are specific challenges and risk areas, which need to be addressed, considered and managed. In this paper we present a list of potential web and distributed risks, which we have identified based on their challenges and characteristics. We survey a number of software risk management approaches and identify their weaknesses and strengths for managing web and distributed development risks. Examples of weaknesses that we identify include the treatment of cultural issues, geographic location, and process and product perspectives. The identified strengths are quite general and only few of them are targeted to web and distributed developments. Following the review of strengths and weaknesses we present an approach called *WeDRisk*, which we propose in order to tackle the weaknesses of the existing approaches, and to accommodate the continuously evolving challenges to web and distributed software development. *WeDRisk* tries to cover some aspects and perspectives, which have not been covered up to now.**

Keywords-software risk management; web development; distributed development; software reliability; WeDRisk approach

I. INTRODUCTION

Software development projects are, by their nature, a risky, complicated and multi-dimensional endeavor [1][2][3][4]. Software risks have been increasing for as long as the software industry has been growing [5]. Many software development projects miss their goals of delivering acceptable software products within agreed constraints of time, budget and quality, due to a combination of the risks themselves, and absent or poor Software Risk Management (SRM) [6][7]. SRM is still evolving, and many software managers have only a limited understanding of its concepts [4]. Industrial risk management practice tends to lag behind recommended risk management best practice, although there are exceptions [4][8][9]. This lag is clearer with Web and Distributed (W-D) software development, where the level of SRM practice is still low.

This paper investigates the abilities of existing SRM approaches to manage W-D software development risks, and to explore their weaknesses, and proposes a novel approach, *WeDRisk*, in order to address the identified weaknesses. The rest of the paper is structured as follows. Section II provides a background on SRM, Section III explores W-D development challenges and their sources of risks. Section IV provides a list of some potential risks to W-D development. We then review the existing SRM approaches (Section V), comparing them based on specific criteria factors (Section VI) in order to investigate their abilities to manage W-D development risks. Section VII introduces the *WeDRisk* approach, which we propose in order to tackle the weaknesses of existing approaches in managing W-D development risks. We then present our conclusions and propose future work in Section VIII.

II. BACKGROUND

This section gives a background of SRM and its related definitions.

A. Software Risk

The Software Engineering Institute (SEI) defines **risk** as "the possibility of suffering loss [10]" and it defines **loss** in a development project, as "the impact to the project, which could be in the form of diminished quality of the end product, increased costs, delayed completion, loss of market share, or failure [10]".

For each risk there are two aspects: risk probability and risk loss. These aspects are used to estimate the impact or Risk Exposure (RE) [11], as follows:

$$RE = P(UO) \cdot L(UO) \quad (1)$$

where,

RE is the Risk Exposure (or risk impact),
P(UO) is the probability of an unsatisfactory outcome, and
L(UO) is the loss associated with an unsatisfactory outcome.

Risk probability estimation is not a straightforward task and can not be 100% accurate (as otherwise there is no risk). Some probability estimation techniques use qualitative data and then convert it into its equivalent quantitative data using some equations, risk-probability table, checklists or relative

scales [6, 11] where some others use subjective Bayesian approach [12] or other techniques.

The top ten software risk items (listed below), which are introduced by Boehm, are examples of sources of risk for software development projects [11].

- Personnel Shortfall
- Unrealistic Schedules and Budget
- Developing wrong software functions
- Developing wrong user interface
- Gold Plating
- Continuing stream of requirements change
- Shortfalls in externally furnished components
- Shortfalls in externally performed tasks
- Real-time Performance Shortfalls
- Straining Computer-science capabilities

A further list of software risk items includes the following risk items [13]:

- Bad traceability
- Insufficient verification and validation
- System complexity
- Customer unsatisfied at project delivery
- Risk reducing technique producing new risk
- Catastrophe/Disaster

Any list of software risk items will need to be updated from time to time, when there are new changes or challenges in software development technology and environment (e.g., social and culture issues, geographically dispersed, new technologies). The significance and type of risks and their sources will also inevitably evolve over time. As an example a recent review [14] found that different authors have identified or proposed different software risks, which means that the number and items of software risks are not fixed. Therefore, new or improved methodologies, techniques and tools to identify, measure and control them are needed.

B. SRM

Boehm [15] defined SRM as "a discipline whose objectives are to identify, address, and eliminate software risk items before they become either a threat to successful software operation or major sources of software rework". Figure 1 shows the basic steps of SRM [11]

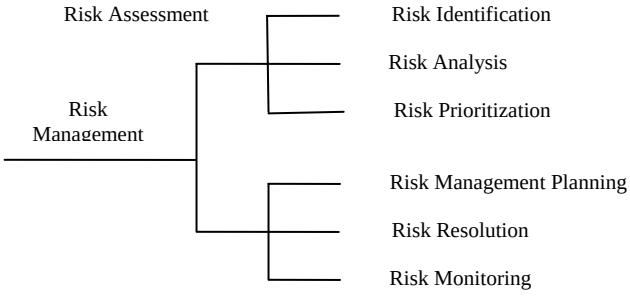

Figure 1. SRM Basic Steps [11]

The main purpose of SRM is to identify potential problems of technical and management aspects before they occur and then take actions to decrease their impact [16].

C. Software Development Perspectives

Software development has three perspectives: project, process and product [17][18]. Looking at these perspectives it is expected that each one of them includes, or could be affected by, different types of risks. For example, the "personnel shortfalls" risk item mainly affects the project perspective, "bad traceability" and "poor testing" affects process whereas "product with wrong functionality" affects product. However, one risk item may affect more than one perspective. Risk management is becoming an important issue from these three perspectives [17][18].

III. CHALLENGES

A number of challenges to traditional software development can be seen in the fields of distributed and web development. The following section focuses on these challenges.

A. Distributed Development Challenges

Distributed Software Development as described by Jimenez and others [19] is a type of development that "allows team members to be located in various remote sites during the software lifecycle, thus making up a network of distant sub-teams". Distributed software projects are usually developed by teams working collaboratively via communication channels (e.g., networks, internet, emails) across many locations. Software developers have adopted distributed software development as a way of reducing the cost and increasing their projects productivity [20]. Developing software across distributed sites presents many challenges, which are summarized in the following points [21][22]:

- Inadequate informal communications
- Lack of trust
- Culture differences (e.g., different language, different corporate culture and different developers' background)
- Time-zone difference (leading to ineffective synchronous communication)
- Development process differences
- Knowledge management challenges (most of the existing management approaches are designed for co-located teams).
- Technical issue: Incompatible data formats and exchanges.
- Security issue (Ensuring electronic transmissions confidentiality and privacy).

All of these challenges could be sources of risk in a variety of development types. In the case of distributed development, they are particularly prevalent challenges and need to be considered by any proposed risk management approach.

B. Web Development Challenges

Web applications are a typical example of web developments, which have become a common type of modern software application. Mendes [23] defines a web application as "an application delivered over the Web that combines characteristics of both Web Hypermedia and Web Software application".

Web applications may be deployed instantly worldwide, without any need for installation or upgrading manuals [24]. They are growing very fast compared with the traditional software, which makes them an important part of the business and software industry. High-performance web sites and applications are used widely in business-to-business ecommerce and many types of services as fully functional systems [25][26].

The development, running and deployment environment of web development need to be considered carefully as well as the significance of associated challenges and risks. Features of the W-D environment, such as diversity and rapid change, present new challenges for the developer, manager, and to traditional project management approaches [26][27[28][29]. More effective risk management methods, models and tools should be introduced to tackle the lack of existing approaches to deal with these challenges [9][30][31].

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

- Their impact and significance are different. For example the exposure to security threats is higher in the web [32][33][34]35].
- As web applications may be deployed instantly worldwide [24], their risks can affect a wider range of components and applications simultaneously in a very short period of time.
- Additional risk sources related to W-D environment include communication, culture, diversity and geographical location [36][37][38][39].
- Estimation of risk probability and loss is more difficult because of the involved challenges and relative lack of experience with them.

Ideally, assessment and management of web development risks should be performed during the whole life cycle of the projects [40], but unfortunately, many web developers use a reactive risk strategy (they do not act until something goes wrong). This strategy is insufficient because it makes software projects vulnerable to any type of risks at any time without effective assessment and control [41].

There is no way to avoid risks in W-D development, so (as with other types of risk) the solution is to attempt to manage them. The following section gives an overview on the state of the art of existing SRM approaches and illustrates their strengths and weaknesses.

IV. W-D POTENTIAL RISKS

The challenges and characteristics of W-D development could bring many risks to W-D development. Some potential risks to W-D developments are listed in Table I [19][21][22][23][24][25][27][32][38][39][40][41][42][43][44]. The list of risks is not final, and could be updated when there are any new challenges or environment changes. Any co-located software risks are also considered risks to W-D development, although their impact and significance could be different.

TABLE I. W-D POTENTIAL RISK ITEMS

SN	Risk Item
1	Unfamiliarity with international and foreign contract law
2	Volatile customer requirement
3	Poor documentation
4	Low visibility of project process
5	Inadequate / inappropriate process development
6	Not enough measurement and estimations
7	Lack of security precautions
8	Weaknesses in protection procedures for Intellectual Property rights
9	Vendor feasibility (weaknesses)
10	Insufficient competence
11	Communication failures
12	Poor sites management control
13	Failure to manage user expectations
14	Insufficient project stakeholder involvement
15	Process instability
16	Poor performance
17	Poor UI (rapid changes)
18	Insecure communication channels
19	Inadequate user involvement
20	Difficulties in ongoing support and maintenance
21	Unrealistic estimation of the number of users
22	Differences in the development methodologies and processes
23	Weak or inadequate contracts
24	Complicated development dependencies between project sites
25	Cross cultural differences / influence
26	Poor product functionality
27	Market fluctuations
28	Scalability limitations
29	Poor availability
30	Lack of top management commitment
31	Instability in other project sites
32	Lack of Face-To-Face meetings
33	Lack of Management availability and efficiency
34	Unfamiliarity with customer type
35	Constraints due to time zone differences
36	Culture Influence
37	Not enough experience with the W-D technologies

Another type of risks that could also affect the W-D developments is the atypical risks type. Atypical risks are risks that could not be predicted before they occur.

V. SRM APPROACHES

There are many different SRM approaches. Some of these approaches are named "models" and others are named "frameworks" or "methods", but they have the same target, which is managing software risks.

Existing SRM methods, models and tools are reviewed in this section. Each of the approaches uses some steps, components or techniques, which may be different or have some similarities with other approaches.

A. Existing Approaches

Nine of the existing approaches have been selected for detailed comparison in this study. The selected approaches are the ones that we expect to satisfy the needs of risk management for software industry in the W-D development environment. The approaches were selected because they are dedicated to manage W-D development risks, or related aspects. The compared approaches are described hereafter.

a) DS-RM-Concept:

Distributed Software - Risk Management Concept (DS-RM -Concept) has been designed based on the idea that communication and continuous risk assessment play a vital role in managing the risks. Risk assessment in this approach uses three concepts: reviews for risk identification; snapshots for analysis; and reports for assessment [45].

b) EBIOS Methodology:

Originally the EBIOS (In French: Expression des Besoins et Identification des Objectifs de Sécurité) method was introduced by Central Directorate of Security of Information Systems (DCSSI) in the French government. It is a risk management methodology concentrating on Information Systems Security (ISS) risks. It consists of a set of guidance steps and it is supported with a free open source software tool. The methodology has five phases: Context Study; Security Requirements Checklist; Threats Study; Identification of Security Objectives and Determination of Security Requirements [33][46]. W-D developments are highly vulnerable to security risks and EBIOS methodology is widely used in government and private sectors to manage such type of risks, as it is supported by an open source tool.

C) ProRisk Framework:

ProRisk is an open system where the users can develop, calibrate a choice from published models (templates) or use different models to accommodate their project need. It is a risk management framework for small and large software projects. However, in order to provide project risk factor a detailed analysis of the project is required [47].

d) Riskit Method:

Riskit method is a SRM method introduced by Jyrki Kontio [48]. Figure 2 shows the process diagram of the method, which is designed to provide organized SRM process and to support involvement of all relevant stakeholders in risk management process [49]. The method is provided with analysis graph and it uses a specific ranking technique called Riskit Pareto Ranking Technique, which uses probability and utility loss ranking [50][51].

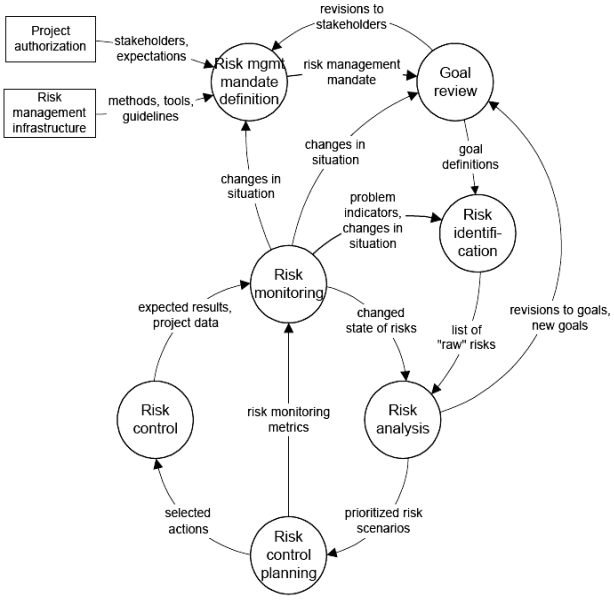

Figure 2. The Process Diagram of Riskit Method [48]

e) SoftRisk:

SoftRisk is model to manage software development risks introduced by the author and others [6]. Figure 3 shows the main steps of SoftRisk model [6][31][52].

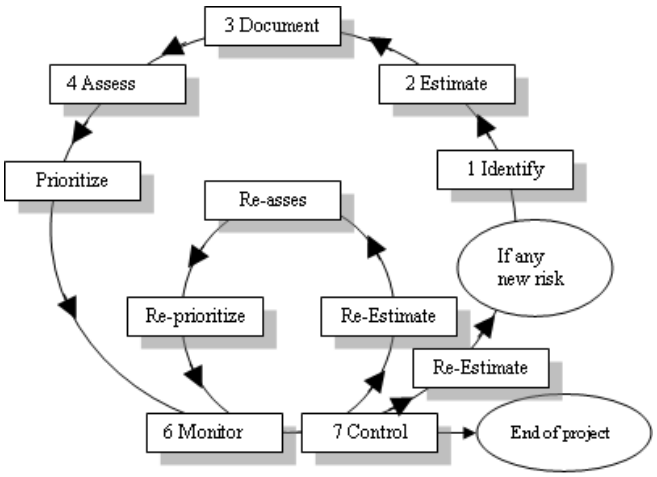

Figure 3. The Main Steps of SoftRisk Model [6]

The model is designed based on the idea of documenting and using historical risk data and focusing on top risks in order to reduce the effort and time in managing software risks. The model has been supported with a prototype tool.

f) CMMI-RSKM:

Capability Maturity Model Integration (CMMI) is an approach for improving processes within organization. The guidance, which is provided by CMMI consists of a group of steps to improve development management, services, and maintenance of products. CMMI has RiSK Management (RSKM) process area and it has been adopted worldwide by many organizations. Its models cover development, acquisition, and services in projects [51][53][54][55].

g) PMBOK RM Process:

Project Management Body of Knowledge (PMBOK) is a process introduced by Project Management Institute (PMI). Its third edition was published in 2004. The PMBOK combines nine areas of knowledge (Integration, scope, time, cost, quality, human resource, communications, purchase and risk). It consists of four process phases - Initiating, Planning, Executing, and Closing. It can be considered as standard for Project Management [51][56][57].

h) GDSP RM Framework:

Geographically Distributed Software Projects (GDSPs) is an integrated framework to manage risks in distributed software projects. It emphasizes many aspects, which are shared between GDSPs and web application developments. The idea behind this framework was based on synthesizing some known risks and risk techniques into integrated approaches. GDSPs links resolution techniques into project risk areas [39]. Elements of the framework are illustrated in Figure 4.

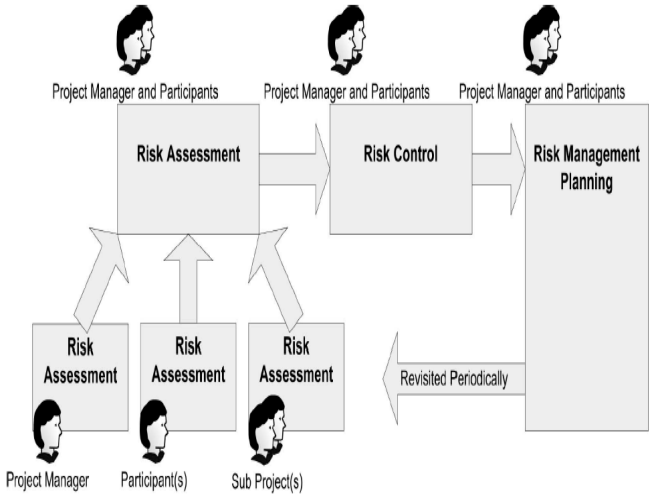

Figure 4. Elements of GDSP's Risks Management Framework [39]

i) Risk and Performance Model:

This model is designed to inspect the relationship between risk and project performance. This includes product and process performance. For this purpose six dimensions (*Organizational Environment*, *User*, *Requirements*, *Project Complexity*, *Planning & Control and Team risk*) of software risks are used by the model [58].

VI. ANALYSIS

The approaches were reviewed for their ability to manage risks of modern software development under the W-D environment and how they can deal with their challenges. In order to see their weaknesses and strengths, a comparison between them has been conducted based on our predefined criteria factors.

The criteria factors were prepared after the challenges, risk areas and characteristics of W-D development were identified, by conducting a risk management practice survey and literature search [16][17][18][20][21][22][23][24][25][28][29][30][31][32][33][34][35][36][37][38][39][40][41]. In order to get a consistent list of criteria factors initially, a list of all criteria factors has been created and then the most related ones to W-D software development were filtered. Meanwhile, some other factors are specified in order to cover aspects that we felt that were not touched before.

The factors cover important risk management aspects (e.g., Perspectives, Communications, Geographically Dispersed, Evolving Environment, Risk Management Evolution, culture issue and Interoperability tracking).

The comparison has been conducted based on available literature such as papers, reports, previous comparison, formal websites of the approaches and related technical reports (references are mentioned in Section V). Table II shows the result of the comparison.

In Table II there are three options for each criteria factor:

- ✔ *when the factor is supported or agreed by the approach.*
- ✕ *if the factor is not supported or not agreed by the approach.*
- **P** *if it is partially supported or partially agreed by the approach.*

Table II can be read either horizontally or vertically. If it is read horizontally then the numbers on the table represent the total of points that each criteria factor has got from all of the approaches for each one of the above three options. If the table is read vertically then the numbers represent the total of points each approach has got for each one of the above three options.

TABLE II. SRM APPROACHES COMPARISON RESULT

Criteria Factors \ Approaches	DS- RM Concept	EBIOS	ProRisk	RisKit	SoftRisk	CMMI-RSKM	PMBOK	GDSP-RM	Risk&Performance	Sub Totals: ✔	✘	P
Perspectives:												
- Project	✔	✘	✔	✔	✔	✔	✔	✔	✔	8	1	
- Process	✘	P	P	P	✘	✔	✔	✘	✔	3	3	3
- Product	✘	P	✘	✘	✘	P	✘	✘	P		6	3
Stakeholder :												
- Involved Stakeholder	✔	P	✔	✔	P	✔	✔	P	P	5		4
- Stakeholder Roles in SRM	P	P	P	P	P	✔	P	P	P	1		8
SRM & Product Quality Link	✘	✘	P	✔	✘	P	✔	✘	✔	3	4	2
Remote SRM	P	✘	P	✘	P	✘	✘	✔	✘	1	5	3
Estimating SRM Cost	✘	✘	P	P	✘	P	P	P	✘		4	5
Provided/Suggested Options :												
- Communications	✔	✔	P	✘	✘	✔	✔	✔	✘	5	3	1
- Collaboration	P	✘	✘	P	✘	✘	✘	P	✘		6	3
Consideration of:												
- Geographically dispersed	✔	✘	✘	✘	P	✘	✘	✔	✘	2	6	1
- Social and legal issues	✘	✔	✘	✘	✘	✘	✘	P	P	1	6	2
- Intellectual property	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
- Ethical issues	✘	✔	✘	✘	✘	✘	✘	✔	✘	2	7	
- Multicultural environment	✘	✘	✘	✘	✘	✘	✘	✔	✘	1	8	
- Evolving environment	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
Preparedness to Atypical Risk	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
Provided SRM Types:												
- Plain	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
- Deep / Ordinary	✔	✔	✔	✔	✔	✔	✔	✔	✔	9		
SRM Evolution Ability	✘	✘	P	✘	✘	✘	✘	✘	✘		8	1
SRM Effect Evaluation	P	✘	P	✔	P	✘	P	P	P	1	2	6
Learning from Mistakes	✔	✘	✘	✘	✔	✘	P	P	✘	2	5	2
Performance Evaluation	P	✘	P	✔	P	✔	✘	P	✔	3	2	4
Acceptable Levels	✘	✘	P	P	✔	✔	✔	✔	✘	4	3	2
Risks of SRM Exploration	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
Prediction Techniques	✔	P	P	✔	P	✘	✘	P	P	2	2	5
Side Affect Absorber	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
Interoperability Tracking	✘	✘	✘	P	✘	✘	✘	P	✔	1	6	2
Dependences Tracking	P	✘	P	P	✘	✘	P	P	P		3	6
Virtual SRM support	P	✘	✘	✘	P	✘	✘	✔	✘	1	6	2
Standard Operation Procedures	✘	✘	✘	✘	✘	✘	✘	✘	✘		9	
Risk Source Tracing	✘	✘	✘	✘	✘	P	✔	✘	✘	1	7	1
Totals :												
✔ Supported or agree	7	4	3	7	4	8	8	9	6	56		
✘ Not Supported or not agree	18	23	17	18	20	20	19	12	19	166		
P Partially Supported or partially agree	7	5	12	7	8	4	5	11	7	66		
Total:										288		

From the numbers that appear in Table II it can be noticed that the total number of criteria factors that are supported or agreed by the approaches has got 56 points from the total of points, which is 288 (with percentage 19%) The ones that are partially supported or partially agree have got 66 points (with percentage from the total of points 23 %) whereas the factors that have got the lowest support by the existing approaches have got the highest number of points, 166 (with percentage 58%). The criteria factors that have got the lowest support are:

- Covering of process and product perspectives
- Consideration of: Geographically dispersed, Social and legal issues, Intellectual property, Ethical issues, Multicultural environment and Evolving environment
- Preparedness for atypical risks
- Plain risk management type
- Evolution of SRM processes
- Exploration of SRM Risks itself

- Risks side affects absorber mechanism
- Risks interoperability tracking
- Standard Operation Procedures

As can be seen in Table II, the points are different from one approach to another. This means that a weak aspect in one approach could be a strong aspect in another one. This is clear from the total points at the end of each approach. On the other hand there are many similarities between many approaches in many aspects as they have the same selections for some criteria factors. Table III summarizes the main strengths and weaknesses of the approaches, from the W-D point of view.

TABLE III. SOME STRENGTHS AND WEAKNESSES OF EACH APPROACH

Approach	Strengths	Weaknesses
DS-RM-Concept	• Targeting of distributed software. • Focusing on communications role. • Supports the use of risks database. • Supported with Risk Guide tool. • It has an effective identification technique.	• It does not consider some aspects such as social, multicultural, and evolving environment. • Lack of risk controlling. • It does not link risk management to development processes and product.
EBIOS Methodology	• Supported with an open source tool. • Its consideration of technical entities and non-technical entities. • Compliance with some standards (ISO27001:2005).	• It is dedicated and limited to Information Systems Security (ISS) risks only. • It has a very limited ability to consider aspects of W-D development environment.
ProRisk Framework	• Can be applied to small and complex projects. • It is open system. • It links business domain to risk management. • It is partially considers the cost in risk management.	• It requires detailed risk analysis. • It depends on other models to perform the risk analysis, which sometimes are not validated enough or not available to the users. • It does not consider most of aspects that are related to W-D environment.
Riskit Method	• It provides conceptual and graphical tool. • It defines project goals based on certain steps.	• It is not supported with risk communication channel. • Other weaknesses can be seen in Table II.
SoftRisk	• It supports risk documentation. • It switches between qualitative and quantitative data. • It is provided with checklist for risks estimation.	• It does not support risk communication. • It does not provide management for product perspective. • Other weaknesses can be seen in Table II.
CMMI-RSKM	• It supports the standardizations in risk management. • It is provided with a sort of guidelines.	• It supports only heavy risk management. • Project managers play most of risk management role. • Many aspects that are related to W-D environment are not considered.
PMBOK RM Process	• It considers the processes of software development. • It includes risk management as a part of project management.	• It is generic to meet some special needs of software projects. • Project managers play most of risk management role. • It does not support many features related to W-D development like consideration of remote risk management, social issues.
GDPS RM Framework	• Consideration of geographically dispersed • It supports categorization of risk areas, risk factors and resolution techniques.	• It uses a predefined list of risk areas and factors, which limits risk identification process. • No integration between risk management and overall project plan. • It does not consider process and product perspectives. • It provides only one type of management. It does not provide plain management.
Risk and Performance Model	• It comes with six dimensions of software risks. • It treats the relation between risk and performance.	• It does not give guidelines for managing risks. • It considers only internal risks. • Other weaknesses can be extracted from Table II.

In general, the associated weaknesses of existing approaches that have resulted from the comparison can be summarized in the following points:

- The existing approaches concentrate on project perspective of software development and they do not pay enough attention to other perspectives (Process and Product).
- They do not accommodate the continuous evolvement and changes issues of software industry and they do not consider aspects related to web, and distributed development environment

(e.g., geographical difference, time zones differences, intellectual property, culture issues, evolving environment etc.).
- Lack of preparedness to atypical risks (No absorbing mechanism for side affects of atypical risks).
- They do not suggest any effective mechanisms to monitor or trace risks interoperability and dependences.
- They are not flexible enough and they offer only deep type of risk management. Plain risk management is not offered.
- Not enough monitoring to SRM performance and its associated risks.
- Most of the approaches are focused on theoretical aspects and do not provide clear guidelines for practicing.

VII. WEDRISK APPROACH

WeDRisk is an approach we propose in order to tackle the weaknesses of existing SRM approaches with more emphasis on W-D development. While the approach is particularly aimed towards W-D development, it should be applicable to modern software developments in general. The general principles of this approach are:

- It is built to tackle the weaknesses of existing approaches, with some new improvements.
- It focuses on W-D software development, but it can be used for others.
- The approach is supposed to be flexible and able to evolve if need be.
- It considers risks from three perspectives (project, process, and product) and uses a modular approach structure of components, phases and layers to manage the complexity in the range of different weaknesses identified.

A. WeDRisk Structure:

The *WeDRisk* approach consists of five layers (Project Layer, Stakeholder Layer, Risk Management (RM) Customization Layer, RM Implementation Layer and Evaluation & Auditing Layer) and two supporter components (Communication & Plug-In Controller and RM Evolution Regulator).

The layers consist of components, which contain steps, techniques and guidelines. The supporter components provide the necessary support to the other *WeDRisk* components. Figure 5 illustrates the main architecture design of *WeDRisk* approach.

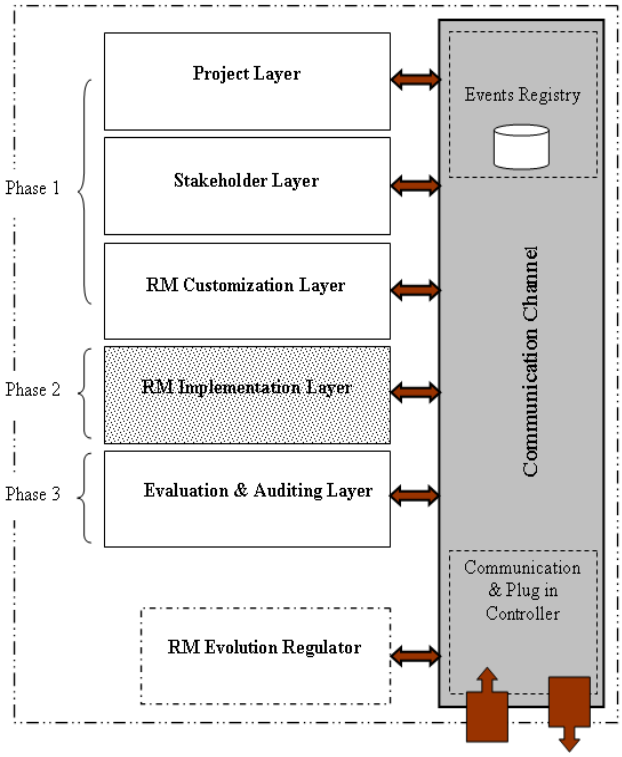

Figure 5. *WeDRisk* Main Architecture

This modular approach structure simplifies the *WeDRisk* design and makes it ready for evolving and integrating.

B. WeDRisk Run Phases & Layer Descriptions

Running phases of the *WeDRisk* consist of three main phases (see Figure 6). They are briefly described below with the appropriate layers that work under them.

First Phase: Establishing RM set-up:

This is an essential phase for RM establishment (set-up). It produces projects' and stakeholders' cards. As well as it customizes the type of RM (deep or plain type). The following layers work under this phase:

Project Layer	**P**roduces /updates Project Card
Stakeholder Layer	**P**roduces Stakeholders Cards
RM Customization Layer	**S**pecifies Management Type

Table IV shows an example of the project card. The data that is shown in the example is dummy data (not real).

TABLE IV. PROJECT CARD EXAMPLE (DUMMY)

Project ID	WP-09-001					
Opening Type	New Project (*may be updating an old one*)					
Project Name	Billing System for van hiring system					
Type	Web Application					
Customer	Newcastle Group					
Project Developer	Advanced SoftGroup Ltd.					
Project Manager	ALI					
Development Sites	One site ; Main Site (Newcastle)					
Development Team	3 Programmers + Editor + Graphics Designer					
Dev. Team Leader	John					
Planned Starting Date	01/04/2009	Planned Finishing Date:	30/04/2009			
Actual Starting Date	05/04/2009	Actual Finishing Data:	25/05/2009			
Initial Contract Cost	£100,000	Actual Cost at Delivery:	£177,000			
Requirement Specification Doc. File	WP-09-001-Req.Pdf					
Events Registry Ref. No	WP-09-001EventReg					
Dependency or Linked Projects	WP-09-201; DP-09-30					
All Project's Identified Risks						
Risk ID	Associated Loss	Responsible	Attack Date	Resolve Date	Attack TREV	Resolve TREV
R-Cu-011	5 days delay	Project Secretary	01/04/2009	04/04/2009	5.7	20.2
~	~	~	~	~	~	~
~	~	~	~	~	~	~
~	~	~	~	~	~	~
R-Cu-034	£500	Site 2 manager	12/05/2009	11/05/2009	4.5	10.5
Project's Current Identified Risks (Prioritized based on TREV)						
Risk ID	Associated Loss	Responsible	Attack Date	Prob.	Mag.	Attack TREV
R-Cu-011	Extra Cost	Programmer No. 1	01/04/2009	0.3	200	60.7
R-Te~	~	~	~	~	~	~
~	~	~	~	~	~	~
~	~	~	~	~	~	~
R-Ge-231						45.5

Second Phase: Implementation of RM Cycles:

The main RM operation/steps are implemented at this phase. The operations include the estimation, evaluation, planning and controlling of the risks.

At this phase risk cards (Table V shows a dummy example of the risk card) are produced for new risks. These cards contain all important identification data of the risks. The identified risks are clustered from their perspectives (Project, Process and Product). Project cards are continuously updated with current risks data. In case of any attack from atypical risks the absorber mechanism will be triggered. Extracting Learned Lessons and tracing dependencies and interoperability are also operations implemented under this phase using special components. The layer that works under this phase is the implementation layer.

Implementation Layer	**P**roduces Risks Cards; Estimates, evaluates, plans for and controls the risks; Deals with atypical risks and cluster risks from the three perspectives (Project, Process and Product), Extracts Learned Lessons traces risks dependencies & interoperability

Third Phase: Evaluation and Auditing

This phase is concerned with RM performance and RM cost evaluation. This required data is periodically collected about RM progression during RM cycle. Collected data cover RM Establishing cost, RM Running cost, Risks Consequences cost, RM durations time and RM efficiency. These data are used to monitor cost and performance of RM operations and it is used to produce RM performance report. It is also used to support evolution of the approach. The responsible layer for this phase is the Evaluation and Auditing Layer.

Evaluation & Auditing Layer	**M**onitors RM progress **P**roduces Performance Report

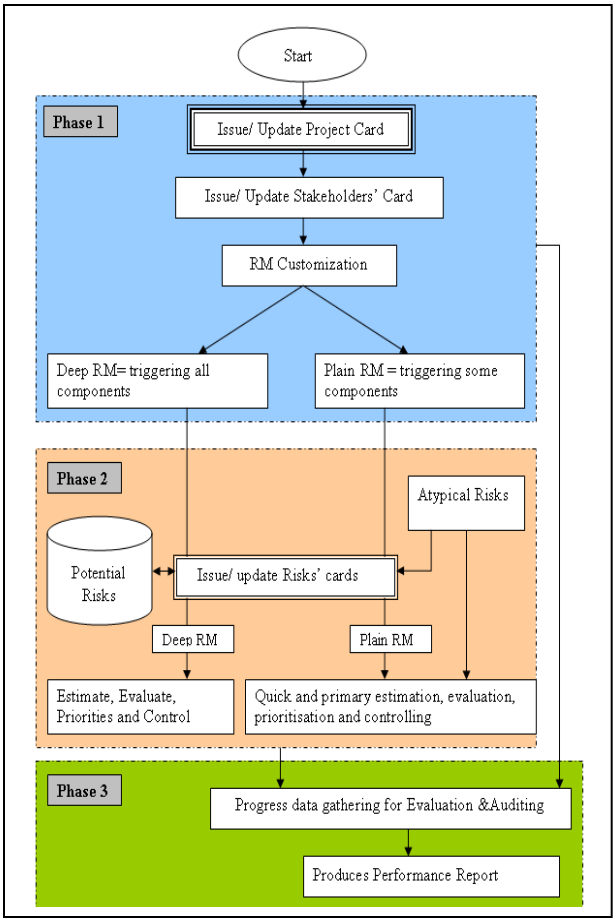

Figure 6: *WeDRisk* Running Phases

C. *WeDRisk Supporter Units*

The role of these units is to provide required support and services to *WeDRisk* components. There are two main units (Communication & Plug-in Controller and RM Evolution Regulator), which are described briefly hereafter:

Communication & Plug-in Controller**:**

The communication and Plug-in Controller works with all layers and at all phases. It consists of the following components:

Communications channel is a component used to ensure all needed communications between all RM layers during the RM cycle. Furthermore, this channel also ensures the exchanges of the RM data between all project sites.

Events Registry is a component used to record some important data about all RM events and actions during the RM cycle. The recorded data are considered as the history of RM implementation cycles, which could be used for statistics, performance monitoring or taking corrective actions.

Plug-in Components provides the support for connecting *WeDRisk* with other approaches. For that it provides standard format for data exchange and checks the permissions and authentications.

Communications Channel	**E**nsures communication for RM operations
Events Registry	**R**egisters RM process Events
Plug-in Component	**L**ink / communicate with others

RM Evolution Regulator**:**

The *Evolution Regulator* is responsible for making any evolutions (improvement or changes) to the RM process. The evolutions are based on needs, enhancement or in some cases as part of corrective actions, which are collected in a special repository called the Evolution Box, and then implemented after they get approval from Evolutions Approval Board. Evolution Regulator components and their roles are summarized below:

Evolution Box	Repository collects data about needs, problems and any evolution suggestions during the RM process
Evolution Approving Board	Evaluates the needs and take evolution decisions; specifies the modifications, priorities, responsibilities and schedules

TABLE V. RISK CARD EXAMPLE (DUMMY)

R. ID	R-Te-011
R. Name	Not enough experience with the W-D technologies
R. source	Programmer
Aspect	Technical Risks
Perspective	Process
Risk Description	The programmer supposed to have enough experience with Java and web services, but he has got stuck with some critical web services aspects.
Risk Factors	- The time is too short to learn web services. - Not enough time/budget to hire programmers - Not enough experience
Potential Impact	Extra Cost (e.g., it cost £3000 per a day for any delays)
Potential Affected Areas	Web related aspects
Dependability of Risks	Testing phase, product perspective
Mitigation Steps	Fast training course, postponing web service part, changing the type of the application or hiring programmer
Primary Precautions	Allocate some funds for hiring extra programmers
Controlling Steps	Hire extra programmers if the time is short, but if there is enough time and less dependency train the existing programmers.
Card Issue Date	18/11/2009
Relation Pointer (Linker)	In our case is null, which means that there is no any risk linked to this risk

D. WeDRisk Distinctiveness

WeDRisk tackles the existing approaches weaknesses by providing new components and covering new aspects to improve RM in W-D development. Although the *WeDRisk* approach is mainly designed for W-D development, it can also be used in the rest of software developments in general. The main contributions of *WeDRisk* are:

- It considers the three W-D perspectives (Project, Process and Product) as it clusters the risks from these three perspectives. This saves time and effort and increases the effectiveness of RM in W-D developments by making the concentration only on the risks of the appointed perspective.
- It provides an absorption mechanism to deal with W-D atypical risks.
- It considers the challenges and characteristics of W-D development since it provides a list of the potential risks that are associated with these characteristics and challenges. This helps to identify current risks faster and easier. This list of potential risks is updateable based on the current challenges and environment.
- The nature of W-D developments needs a flexible RM, therefore *WeDRisk* approach offers two types of RM (plain and deep).
- *WeDRisk* has been provided with an Events Registry component, which works as a log file, recording important events data during RM operation progression.
- Communication plays a vital role in managing W-D development risks. Therefore, the approach has a Communication and Plug-in Controller to ensure the internal communication (between approach components) via a communication channel, and external communication with other approaches via Plug-in unit.
- The approach includes W-D factors as a part of the risk estimation equation.
- Risks network is very complicated in W-D development projects. Combination of some risks could produce new risks or increase their severity. Meanwhile as many projects are multisite projects there is a dependency among them. *WeDRisk* treats this with a special component called Dependencies & Interoperability Tracer.
- *WeDRisk* is an evolutional approach as it has been designed to accommodate the evolutions in W-D developments.

E. Benchmarking

Comparing with other approaches, *WeDRisk* maintains the strengths of existing approaches and tackles their weaknesses in managing W-D risks. It designed to be an evolutionary approach. Table VI illustrates how *WeDRisk* approach comes with new features to improve the RM in W-D development.

TABLE VI. BENCHMARKING TABLE

Current Approaches	*WeDRisk* Approach
Perspectives Consideration	
The consideration is mainly on Project Perspective	It considers all perspective (project Product Process) and clusters the risks from all perspectives.
Evolution Ability	
They are fixed approaches	It is flexible to accommodate the W-D evolutions. It has a special component to handle that.
Offered RM types	
Usually they offer one type of RM, which is Deep RM type.	*WeDRisk* offers two types of RM (Deep and Plain). RM can be customized based on the situation needs, availability of resources and criticality of time.
Preparedness to atypical Risk	
None of them can deal with to atypical risk	*WeDRisks* has a mechanism to deal with atypical risks
W-D risk estimation and assessment	
Not enough consideration to W-D factors	It includes W-D factors at risks estimation equations
Dependencies & Interoperability	
Very limited and indirect ability	*WeDRisk* maps risks Dependencies and Interoperability
Auditing and Evaluation	
Limited in some of them	It has components for RM cost and performance evaluation
Learning from Mistakes	
Somewhat some of them have databases that can be used to learn from previous cycles	*WeDRisk* Extracts Learned Lessons from RM cycles
Communication	
Some of them has good communication channels	It has a communication channel supported with events registry and plug-in components

F. WeDRisk Evaluation

Currently, we are in the stage of evaluating the *WeDRisk* approach. For the evaluation purpose we have planned for two options, which are:

- Evaluating the whole approach (all components together) using one or more case studies.
- Dividing the approach into 'chunks' (representing the novel aspects in the *WeDRisk* approach) and then evaluating them one by one using case studies or experiments.

The preparation for the two options is currently in progress. A case study has been designed for the first option whereas, for the second option an experiment has been designed to evaluate the first 'chunk', which covers:

- Proposed list of W-D potential risks
- The usefulness/effectiveness of clustering the risks from the three perspectives (project, process product) and clustering criteria
- Potential of atypical risks in W-D development

Due to some difficulties in getting suitable projects where a case study can be executed, we plan to start with the second option (the experiment). The subjects for the first experiment are PhD and MSc students at School of Computing Science, Newcastle University, UK. The experiment was conducted in July/August 2010.

Other experiments or case studies will be designed to evaluate other novel aspects of *WeDRisk* approach and the evaluation results will be presented in the forthcoming papers.

G. The Prototype

It is expected that *WeDRisk* approach will lead to the development of a risk management tool. The tool will be targeted for use by W-D software development houses to manage W-D development risks. The tool functions are intended to comply with the proposed *WeDRisk* components and techniques. Currently the prototype of the tool is under construction. There are some challenges for the prototype implementation, which include:

- The prototype should cover important novel aspects of *WeDRisk*.
- It should be a web application and be able to deal with W-D multisite developments.
- It should be able to cover the three perspectives (project, process and product).
- It should be supported with a database for risks, projects and stakeholders cards data.

Implementing the prototype early could help in *WeDRisk* evaluation. However, in order to reduce rework in the implementation, commencing work on the prototype is dependent upon the completion of the first phase of the *WeDRisk* evaluation. We expect that the prototype could accelerate the rest of evaluation and validation phases and saves the time and effort. Moreover the result of evaluation can be considered as evaluation for both *WeDRisk* and the prototype, and can be used to improve both of them and to build a reliable tool based on the prototype. The work finished in the prototype implementation includes the design and creation of the supported database and building some main components. The prototype is expected to be ready in the middle of 2011.

VIII. Conclusion and Future Work

The paper has identified the challenges of W-D development and shown how the importance of risk in this context is different from others. A list of potential risks to W-D has been presented. The list is just an initial one, and should be updated from time to time when there are any new challenges or changes in the development environment. In order to investigate the weaknesses and strengths of existing approaches in managing the risks in W-D, the related existing SRM approaches have been reviewed and compared. The comparison is based on special criteria factors, which are prepared carefully in order to examine the ability of the approaches to manage the risks of W-D software development. The weaknesses and strengths of the compared approaches are identified in this paper. In general, most of the identified strengths are related to co-located development software and they are spread among the approaches.

It can be concluded that though there are many SRM approaches there is still a large gap between the existing approaches and actual practicing in software industry practice. This is due to the associated weaknesses in the approaches (e.g., not enough consideration to: difference in geographical locations, culture issues, process perspective and product perspective).

From Table II and Table III the following points can be concluded:

- There is no single approach that is able to manage software risks in W-D environments alone, unfortunately the strengths of the approaches are dispersed between them. In the current situation the developers either have to use more than one approach or miss some aspects and support.
- Tackling the weaknesses of the approaches and combining the strengths of them in a new approach is a step toward improving risk management in W-D environment.

For effective risk management in W-D development all challenges, characteristics, risk areas, development and running environment and development perspectives (project, process and product) and other related aspects must be considered.

The reviewed approaches have added significant value to traditional software development projects, but it is clear that the W-D developments are not yet well covered.

As a part of ongoing PhD research at School of Computing Science, Newcastle University, UK, the *WeDRisk* approach to manage W-D development projects risks has been presented in this paper. The approach aims to tackle the weaknesses in existing approaches and to propose new management concepts in order to improve the level of practicing of SRM in the field. While the approach is particularly aimed towards W-D development, it should be applicable to modern software developments in general.

The *WeDRisk* approach has been designed to satisfy the needs of risk management for W-D development. *WeDRisk* provides some contributions to manage W-D risks such as: the consideration of W-D risks from three perspectives (Project Process and product); involving specific factors for W-D as a part of risks estimation equations; providing a mechanism to deal with atypical risks; ability to evolve; mapping the dependencies and interoperability of the risks; managing risks across multisite projects; and reflecting W-D risks by providing an updateable list of W-D potential risks. In addition to W-D development *WeDRisk* is thought to be ready for serving other software development. The future work in this project includes more evaluation of *WeDRisk* and completing the prototype tool.

REFERENCES

[1] A. Keshlaf and S. Riddle, "Risk Management for Web and Distributed Software Development Projects," in 2010 Fifth International Conference on Internet Monitoring and Protection (ICIMP 2010), IEEE Computer Society, 2010, pp. 22-28.

[2] Y. Kwak and J. Stoddard, "Project Risk Management: Lessons Learned from Software Development Environment," Technovation, vol. 24, November 2004, pp. 915–920.

[3] H. Yong, C. Juhua, R. Zhenbang, M. Liu, and X. Kang, "A Neural Networks Approach for Software Risk Analysis," Proc. of Sixth IEEE International Conference on Data Mining - Workshops (ICDMW'06), IEEE, 2006, pp. 722-725.

[4] P. Bannerman, "Risk and Risk Management in Software Projects: A Reassessment," Journal of Systems and Software - Elsevier, vol. 81, December 2008, pp. 2118 - 2133.

[5] A. Tiwana and M. Keil, "Functionality Risk in Information Systems Development: An Empirical Investigation," IEEE Transactions on Engineering Management, vol. 53, AUGUST 2006, pp. 412- 425.

[6] A. Keshlaf and K. Hashim, "A Model and Prototype Tool to Manage Software Risks," Proc. of First Asia–Pacific Conference on Quality Software, IEEE Computer Society, 2000, pp. 297–305.

[7] S. Islam, "Software Development Risk Management Model – A Goal Driven Approach," Proc. of ESEC/FSE Doctoral Symposium'09, ACM, 2009, pp. 5-8.

[8] J. Esteves, J. Pastor, N. Rodriguez, and R. Roy, "Implementing and Improving the SEI Risk Management Method in a University Software Project," Latin America Transactions, IEEE (Revista IEEE America Latina), vol. 3, March 2005, pp. 90-97.

[9] M. Kajko-Mattsson and J. Nyfjord, "State of Software Risk Management Practice," IAENG International Journal of Computer Science - On-line Issue, vol. 35, November 2008.

[10] R. Williams, G. Pandelios, and S. Behrens, "Software Risk Evaluation (SRE) Method Description (Version 2.0)," Software Engineering Institute (SEI) December 1999.

[11] B. W. Boehm, "Software Risk Management: Principles and Practices," IEEE Software, vol. 8, 1991, pp. 32-41.

[12] J. Moses, "Bayesian Probability Distributions for Assessing Measurement of Subjective Software Attributes," Information and Software Technology, vol. 42, 15 May 2000, pp. 533-546.

[13] K. Hashim and A. Keshlaf, "An Approach to Sharing Solutions to Software Project Management Problems," Proc. of International Conference on Information Management and Engineering (ICIME '09), IEEE Computer Society, 2009, pp. 694-697.

[14] W. Han and S. Huang, "An Empirical Analysis of Risk Components and Performance on Software Projects," Journal of Systems and Software, vol. 80, January 2007, pp. 42-50.

[15] B. W. Boehm, Software Risk Management, IEEE Computer Society Press, 1989.

[16] IEEE Std. 1540-2001, "IEEE Standard for Software Life Cycle Processes – Risk Management," IEEE 2001.

[17] B. Boehm, J. Kwan, D. Port, A. Shah, and R. Madachy, "Using the WinWin Spiral Model: A Case Study," IEEE Computer, 1998, pp. 33 - 44.

[18] S. Misra, U. Kumar, V. Kumar, and M. Shareef, "Risk Management Models in Software Engineering," International Journal of Process Management and Benchmarking (IJPMB), vol. 2, 2007, pp. 59-70.

[19] M. Jiménez, M. Piattini, and A. Vizcaíno, "Challenges and Improvements in Distributed Software Development: A Systematic Review," Advances in Software Engineering, vol.2009, 2009, pp. 1-14.

[20] M. Malarvannan, "Managing Offshore Software Teams." vol. 2009: Outsource Portfolio, 2009.

[21] B. Sengupta, S. Chandra, and V. Sinha, "A Research Agenda for Distributed Software Development," Proc. of 28th International Conference on Software Engineering (ICSE'06), ACM, 2006, pp. 731 - 740.

[22] K. Nidiffer and D. Dolan, "Evolving Distributed Project Management," IEEE Software, vol. 22, September/October 2005, pp. 63-72.

[23] E. Mendes and N. Mosley, Web Engineering, Berlin Heidelberg: Springer-Verlag, 2006.

[24] A. Taivalsaari, "Mashware: The Future of Web Applications." vol. 2009: Sun Microsystems Laboratories, 2009.

[25] J. Offut, "Quality Attributes of Web Software Applications," IEEE Software, vol. 19, March / April 2002, pp. 25-32.

[26] F. Donini, M. Mongiello, M. Ruta, and M. Totaro, "A Model Checking-based Method for Verifying Web Application Design," Electronic Notes in Theoretical Computer Science, vol. 151, 31 May 2006, pp. 19 - 32.

[27] C. Beise, "IT Project Management and Virtual Teams", Proc. of (SIGMIS'04) Conference Arizona, USA,Tucson, 2004, pp. 129-133.

[28] B. Behkamal, M. Kahani, and M. Akbari, "Customizing ISO 9126 Quality Model for Evaluation of B2B Applications," Information and Software Technology, vol. 51, March 2009, pp. 599-609.

[29] J. Tian, S. Rudrarjuand, and Z. Li, "Evaluating Web Software Reliability Based on Workload and Failure Data Extracted from Server Logs," IEEE Transaction on Software Engineering, vol. 30, November 2004, pp. 754 - 769.

[30] J. Kontio, M. Hoglund, J. Ryden, and P. Abrahamsson, "Managing Commitment and Risks: Challenging in Distributed Agile Development," Proc. of 26th International Conference on Software Engineering (ICSE '04), 2004, pp. 732- 733.

[31] M. Rabbi and K. Mannan, "A Review of Software Risk Management for Selection of Best Tools and Techniques," Proc. 9th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel / Distributed Computing, IEEE Computer Society, 2008, pp. 773 - 778.

[32] W. Glisson and R. Welland, "Web Development Evolution: The Assimilation of Web Engineering Security," in Proc. of Third Latin American Web Congress (LA-WEB'05), IEEE Computer Society, 2005, p. 5.

[33] B. Romero, H. Haddad, and J. Molero A, "A Methodological Tool for Asset Identification in Web Applications Security Risk Assessment," in Proc. of Fourth International Conference on Software Engineering Advances, IEEE Computer Society, 2009, pp. 413-418.

[34] Y. Huang, C. Tsai, D. Lee, and S. Kuo, "Non-Detrimental Web Application Security Scanning," in Proc. of the 15th International Symposium on Software Reliability Engineering (ISSRE'04): IEEE Computer Society, 2004, pp. 219-239.

[35] X. Ge, R. Paige, F. Polack, H. Chivers, and P. Brooke, "Agile Development of Secure Web Applications," Proc. of 6th International Conference on Web Engineering (ICWE'06), ACM, 2006, pp. 305 -312.

[36] CA/Wily, "White Paper: Effectively Managing High-Performing Business-Critical Web Application," http://zones.computerworld.com/ca/ accessed on 16 Jan. 2011.

[37] G. Kappel, B. Proll, S. Reich, and W. Retschitzegger, Web Engineering the Discipline of Systematic Development of Web Application, John Wiley & Sons, Ltd, 2006.

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

[39] J. Presson, L. Mathiassen, B. Jesper, T. Madsen, and F. Steinson, "Managing Risks in Distributed Software Projects: An Integrative Framework," IEEE Transaction on Software Engineering, vol. 56, 2009, pp. 508-532.

[40] S. Willis, Using QA for Risk Management in Web Projects, Software Quality and Software Testing in Internet Times, New York,USA: Springer-Veriag Inc., 2002.

[41] P. Pressman and D. Low, Web Engineering A Practitioner's Approach. International Edition, Mc Graw Hill, 2009.

[42] C. Iacovou and R. Nakatsu, "A risk profile of offshore-outsourced development projects," Communications of the ACM, vol. 51, 2008, pp. 89-94.

[43] R. Prikladnicki, J. Audy, and R. Evaristo, "Global Software Development in Practice Lessons Learned ," Software Process Improvement and Practice, vol. 8, October/December 2003, pp. 267-281.

[44] C. Ebert, B. Murthy, and N. Jha, "Managing Risks in Global Software Engineering: Principles and Practices," in Proc. of 2008 IEEE International Conference on Global Software Engineering. 2008, pp. 131-140.

[45] J. Gorski and J. Miler, "Towards an Integrated Environment for Risk Management in Distributed Software Projects," Proc. of 7th European Conference on Software Quality (ECSQ02), Helsinki, Finland, 2002.

[46] ENISA, "Ebios Product Identity Card," ENISA, http://rm-inv.enisa.europa.eu/methods_tools/m_ebios.html accessed on 28 December 2010.

[47] G. Roy, "A Risk Management Framework for Software Engineering Practice," in Proc. of the 2004 Australian Software Engineering Conference (ASWEC'04), IEEE Computer Society, 2004, pp. 60-67.

[48] J. Kontio, "The Riskit Method for Software Risk Management, Version 1.00 CS-TR-3782 / UMIACS-TR- 97-38," University of Maryland, Maryland 1997.

[49] J. Kontio and V. R. Basili, "Empirical Evaluation of a Risk Management Method," in SEI Conference on Risk Management Atlantic City, Nj, USA, 1997.

[50] B. Freimut, S. Hartkopf, P. Kaiser, J. Kontio, and W. Kobitzsch, "An Industrial Case Study of Implementing Software Risk Management," in Proc. of 8th European Software Engineering Conference held jointly with 9th ACM (SIGSOFT) International Symposium on Foundations of Software Engineering, ACM, 2001, pp. 277- 287.

[51] J. Dhlamini, I. Nhamu, and A. Kachepa, "Intelligent Risk Management Tools for Software Development," in Proc. of the 2009 Annual Conference of the Southern African Computer Lecturers' Association (SACLA 09), ACM, 2009, pp. 33 - 40.

[52] J. Smith, S. Bohner, and D. McCricard, "Project Management for the 21st Century Supporting Collaborative Design Through Risk Analysis," Proc. of 43rd ACM Southeast Conference, ACM, 2005, pp. 300 - 305.

[53] C. Pan and Y. Chen, "An Optimization Model of CMMI-Based Software Project Risk Response Planning," International Journal of Applied Mathematics and Computer Sciences, vol. 1, 2005, pp. 155 - 159.

[54] SEI-CMMI, "CMMI-SVC,V1.2." Software Engineering Institute, http://www.sei.cmu.edu/reports/09tr001.pdf accessed on 30 December 2010.

[55] R. Williams, "The CMMI RSKM Process Area as a Risk Management Standard," in Proc. of Sixteenth Annual International Symposium of the International Council On Systems Engineering (INCOSE): INCOSE, 2006.

[56] D. Callegari and R. Bastos, "Project Management and Software Development Processes: Integrating RUP and PMBOK," in Proc. of 2007 International Conference on Systems Engineering and Modeling, IEEE, 2007, pp. 1- 8.

[57] W. R. Duncan, "A Guide to the Project Management Body of Knowledge PMBOK- PMI," Project management Institute, Boulevard -Newtown Square, USA 1996.

[58] L. Wallace, M. Keill, and A. Rai, "How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model," Decision Sciences vol. 35, 2004, pp. 289 – 321.