A Risk Management Tool for Extreme Programming

Hassan Mathkour, Ghazy Assassa, and A. Baihan

Department of Computer Science, King Saud University, Riyadh, Saudi Arabia

Abstract
Risk management is the application of appropriate tools and procedures to contain risk within acceptable limits. In this paper, the authors apply risk management to software development that uses extreme programming approach. A risk tool is designed and developed using MS Excel. The tool is simple to use and would help risk analysis of the twelve practices of extreme programming. The tool has been used and experimented with. Information such as project budget , risk management budget, cost of controls, SLE, ARO have been entered into the tool to analyze the priority practices in order to decide which practice must be dealt with first.

Keywords:
Risk management, extreme programming, ARO, SLE.

1. Introduction

Project risk is defined as the problems, which have not happened yet, that could cause some loss or threaten the success of the project, whereas, risk management is the application of appropriate tools and procedures to contain risk within acceptable limits [20, 18]. In real world, we need to manage risks in order to reduce unexpected and costly surprises, achieve better results from projects and provide better information for decision making. The benefits of risk management include important issues such as financial, marketing and management benefits [11, 17].

With respect to domains, risks are classified as financial risks [3, 17], health risks, environmental risks, security risks [17], commercial risks, operational risks, people risks, organizational risks, compliance risks [3], technical risks [3, 15], strategic risks, tools risks, requirements risks, estimation risks [15], etc. With respect to severity, risks may be classified as acceptable risks, manageable risks, and serious (significant) risks [12]. Other categorization of risks is given as known risks (low risk), unknown known risks (Medium risk), unknown unknowns risks (High risk) (Sugianto, 2002). In [15] is a different view of categorization: project risks, product risks and business risk [15]. The risk management process, Figure 1, includes four basic steps: risk identification, analysis, planning, and monitoring.

This paper is organized as follows: section 2 reviews the existing risk management tools. Section 3 discusses Extreme Programming and Risk. Section 4 presents the details of the proposed risk tool. Section 5 presents some results obtained by the tool. Finally, the conclusion and future work are given in Section 6.

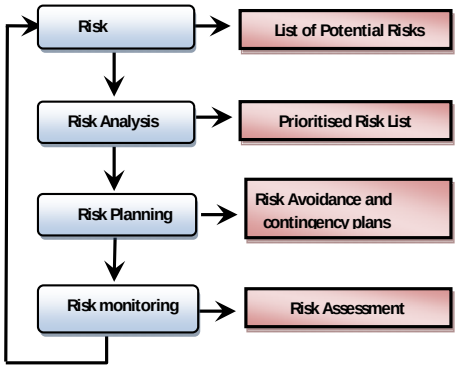

Figure 1: The Risk Management Process [15].

2. Risk Management Tools

This section presents the features and discusses the merits of six risk management tools, namely, Vanguard studio, Pertmaster , @Risk, RiskEase, CrystalBall 7, RiskTrak [4, 10, 13]. Many companies use one or more tool for risk management; however, most of these companies use only 10% to 25 % from each tool's features [16]. Some features of the reviewed tools are selected for inclusion in the proposed risk tool as discussed in section 4.

Vanguard studio

Vanguard is a powerful system used for decision-support analysis and business modeling. It helps making business decisions that are more likely to yield favorable results by providing a rich set of techniques for dealing with risk and planning contingent actions. Features of Vanguard system include general modeling and problem solving, collaborative modeling, data analysis, advanced analytics, forecasting, decision tree analysis, sensitivity analysis, Monte Carlo simulation, optimization, and application development [19].

RiskEase

RiskEase is a state-of-the-art risk analysis tool, which was originally developed under the name "RiskMaster for DOS" and "RiskMaster for Windows" for the Harvard University Program in Investment Appraisal and Management; it has been used since 1987 by many reputable institutions around the world as part of their

Manuscript received August 5, 2008
Manuscript revised August 20, 2008

training courses in cost-benefit analysis. RiskEase built upon the successful design of its predecessors and adds a host of new features that enhance both functionality and ease of use. Features of RiskEase include easy to use point-and-click interface, complete sensitivity analysis module, powerful user interface for editing probability distributions with instant real-time display, ability to bend, truncate and apply correlation conditions to probability distributions, capability to fit probability distributions to user provided data, probability distribution library, optimization within simulation, enhanced analysis module capable of comparing multiple simulation results, ability to use the software in "real options" situations, built-in report generator enabling the generation of professional reports, many other features such as the use of default distributions, new chart formatting options, annotation features and context sensitive on-line help. Details are found in http://www.riskease.com/index.html.

CrystalBall

Crystal Ball 7 Premium Edition is considered as the most complete suite of risk analysis, forecasting, optimization and real options analysis tools available for spreadsheets. With one integrated toolset, user can use his own historical data to build accurate models, automate what-if-analysis to understand the effect of underlying uncertainty, apply real options theory to enhance the value of each project by incorporating strategic flexibility in decision making, and search for the best solution or project mix.

Features of Crystal Ball include Monte Carlo simulation, distribution gallery, categories of distributions , publish and subscribe feature for categories, process capability features, forecast charts, split-view charts, sensitivity and tornado analyses, distribution fitting, correlation, charting and reporting, precision control, Latin hypercube sampling, data extraction ,CB tools, Microsoft certification. Details are in http://www.crystalball.com.

RiskTrak Professional

Project Management software presently available is geared to managing schedules. RiskTrak is a project/program-level tool designed specifically to identify, report and manage risks across an organization, connecting team members over existing network. Unlike other Risk Analysis software tools that's on the market, RiskTrak is an enterprise-wide Risk Management tool. Features of RiskTrak Professional include encrypted polymorphic project database, RiskTrak introductory interview expert, SEI taxonomy interview experts, emu/euro risktrak interview expert, top-level graphical charts, MFC drag & drop, multiple risk mitigations, assessment & detail reports, secure audit trail, clipboard cut & paste. , Context-sensitive F1 Help, SQL engine, export data (*.CSV), attach files to risks and mitigations, networking capability and network notifications, security/user permissions, automated risk management report, direct import of MS Project *.MPX Files, ODBC import and mapping of *.CSV files, split-screen SQL view, export project files to floppy, multi-project queries, extensible database (user-defined fields). Details are found in http://www.risktrak.com/rstdemo.htm.

PertMaster Risk Expert

Features of PertMaster Risk Expert include Monte Carlo analysis, Latin hypercube analysis, optional fixed seed point, risk register, generic risk import/export, hierarchical resources, resource uncertainty, estimate uncertainty, task existence, duration quick risk, resource quick risk, task lag quick risk, risk correlation, schedule sanity check, template quick risk, critical path report, online tutorials, no database is required. Details are in http://www.pertmaster.com.

@ Risk

This tool shows many possible outcomes including seamless integration into Microsoft excel, intuitive toolbars, advanced simulation engine, 37 built-in distribution functions, distribution viewer and data fitting, percentile distribution parameters, correlation of inputs, presentation-quality graphs, one-click quick reports, histograms, area, line, cumulative, summary, overlay, tornado graphs, reporting in excel, sensitivity and scenario analysis.

3. Extreme Programming and Risk

The development of software systems is a risky endeavor that usually encompasses constraints of schedule and budget. Several methodologies are proposed to remedy the problems encountered in traditional development methodologies used in software engineering. Currently, Extreme Programming (XP) is considered as the most famous and prominent agile methodology]5] Extreme Programming is based on twelve best practices [1, 5, 7] of which the following are core practices: Customer On Site, Planning Game, collective Code Ownership, Simple Design. The risk factors that might occur if the Customer is not On Site include changing requirements, cost of communication, decisions not forthcoming, difficulty in communication and incomplete requirements. Similarly, the risk factors that might occur if the Planning Game is not applied include changing priority and bad fit to schedule. The risk elements that might occur if the Code Ownership practice is not applied include architectural degeneration. Also, the risk factors that might occur if the Simple Design practice is not applied include changing requirements [9].

This pilot study [8] included some 25 questions directed to several channels (82 responses) including a group of final-year students of the Computer Science major at the

University of Manitoba (44 students groups), a number of software developers working in the Greater Toronto Area (27 Toronto groups) , and an open call on a number of web sites (10 through the web site links). The study assumed that the respondents had the basic knowledge of XP core values and practices as well as of software development in general. Table 1 is extracted from the pilot study [8] and shows for each of the twelve XP practices the corresponding usefulness mean and standard deviation Mean/SDev as well as the difficulty mean and standard deviation. All values are scaled between 0 and 4, where the value 4 corresponds to the most useful and most difficult. Regarding the twelve XP practices shown in Table 1, most of them were deemed useful or very useful, with simple design and test-driven development being on the top. Besides, most of the practices were deemed easy or very easy to achieve. Simple design was deemed the most difficult, closely followed by pair programming, customer tests, and collective code ownership.

Table 1: Descriptive statistics for opinions about usefulness/relevance and difficulty of achieving the twelve XP practices [8].

XP practice	Usefulness		Difficulty	
	Mean	SDev	Mean	SDev
Continuous integration	3.347	0.7786	1.837	0.8743
Pair programming	3.02	0.8777	2.184	0.9503
Test-driven development	3.551	0.7922	2.000	1.000
Refactoring	3.367	0.8340	1.898	0.8227
Collective ownership	2.959	0.7348	2.102	0.8719
Simple design	3.571	0.5401	2.347	0.8304
Small releases	3.041	1.040	1.796	0.8411
Planning game	2.592	1.353	1.469	0.9811
Customer On Site	3.265	1.114	2.082	0.9755
Metaphor	1.959	1.258	1.592	1.206
Coding standards	3.429	0.7906	1.510	0.7394
Sustained pace	3.224	0.9189	1.980	1.031

4. The Proposed Risk Tool

This section presents the details of the proposed risk tool. The features included in the tool and its design are discussed.

4.1 Features of the proposed risk tool

Table 2 displays a summary of nine features included in the above discussed six tools.

1- Easy to use point-and-click interface: the interface of application is easy to use.
2- Quick Reports: generate summary charts and graphs.
3- Online tutorials: available on the internet.
4- Reporting in Excel: All graphs and charts can be exported to Excel for easy distribution.
5- Sensitivity and scenario analysis module: Identify the individual tasks that have the most impact on results, and the particular scenarios that lead to certain results.
6- Data extraction (to any program): Allows users to examine individual simulation results and transfer results to other software programs.
7- Work dependency: The tool works with support from other programs like MS- Excel, MS-Project, etc.
8- Integration with other programs: The tool can exchange data with one or more programs.
9-Requires Database: The tool needs database to store all risks information like, e.g., risk name, risk priority, etc.

In this paper, a subset of these features is selected for implementation. These are 1, 2, 5, and 7. The last column in table 2 indicates the reason for excluding a particular feature from the proposed risk management tool:

a- No need for online tutorials because the proposed tool is easy to use.
b- The tool is based on MS-Excel; there is no need to report to MS-Excel.
c- There is no need to have other programs because MS-Excel is sufficient for the intended tool.

d- There is no need to store more data as long as MS-Excel is used.

Table2: Features of the proposal risk tool.

Feature #	Vanguard studio	Risk	@Risk	RiskEase	CrystalBall 7	RiskTrak	Proposal	Reason for
1.				√			√	
2.			√		√	√	√	
3.		√						a
4.			√					b
5.	√		√	√	√		√	
6.					√	√		c
7.	√	√		√		√	√	
8.		√						c
9.						√		d

4.2 The design of the proposed Tool

After reviewing the above explained 6 tools, a decision is made to use MS-Excel as it is: easy to use, popular, and covers our needs. In addition, Ms-Excel does not require installation or training.
The tool includes ten columns (Figure 2 below): XP Practices, SLE, ARO, ALE, Loss"prior", Priority, Cost of control, Deal, Loss"after" and Cost of control needed. The first column shows the 12 XP practices and categorizes each as high, medium or low practice. These are also ranked decreasingly according to Shodan survey questions study. Details are in
http://c2.com/cgi/wiki?ShodanInputMetric.

The first column is a sequence numbering of the XP practices. The second column shows SLE "Single Loss Expectancy". It is the weight (out of 100) of any XP Practice in the project. The values of the SLE column may come from one of two sources, local or non local environment. SLE data for local environment may be obtained via an assessment questionnaire; for example, a questionnaire to be distributed to students working on their projects, or form practitioners in software houses. SLE data for non local environment are available in the pilot study of reference [8] that determines the SLE value for each practice under column Mean Usefulness, out of 4. In the current study the values are scaled out of 100 and each practice is given a percentage relative to other twelve using the formula ((Practice value/Total practice value) x 100). For example; reference [8] gives pair programming practice the value 3.020 out of 4 that is converted to 75.5, (3.020 x 25=75.5), and a percentage value of 8.09, ((75.5/933)x100= 8.09). Another example for the planning game practice is given in [8]. It gives the value 2.592 out of 4 that is converted to 64.8, (2.592 x 25=64.8), and percentage value of 6.94, ((25=64.8/933)x100= 6.94). Table 3 shows these converted values.
The third column is ARO "Annualized Rate of Occurrence." It is the range of not applying the practice of XP in a project within a one-year timeframe. The range can be from 0 (never) to 5 (five times a year).

Table 3: Converted mean values of XP practices [8].

XP Practices	Mean (0-4)	Mean (%)
Continuous integration	3.347	8.97
Pair programming	3.02	8.09
Test-driven development	3.551	9.51
Refactoring	3.367	9.02
Collective ownership	2.959	7.93
Simple design	3.571	9.57
Small releases	3.041	8.15
Planning game	2.592	6.94
Customer On Site	3.265	8.75
Metaphor	1.959	5.25
Coding standards	3.429	9.19
Sustained pace	3.224	8.64
Total	**37.325**	**100**

4.3 Calculation of ARO values

In the assessment questionnaire, each group should indicate the range of applying each practice (0-> not applicable ,1-> very low, 2-> low, 3-> medium, 4-> high, 5-> very high) in each day during the project execution. We then get the mean value for each practice during the project execution by dividing by 28 days. For example; the collective ownership practice yields the value 112/28=4 (out of 5). In the above it is assumed that 1 month for students is equivalent to 28days. Different values may be obtained for practitioners in software houses.

The fourth column is ALE “Annualized Loss Expectancy" that yields the overall loss annualized per practice of XP. The values of ALE are obtained from the formula: ALE=SLE x ARO. For example, if the collective ownership XP practice is not applied in the project, this can cause 7.9 in damage and for ARO =1 (project duration of 12 months), then the ALE value is 7.9 x1 = 7.9.
The fifth column is Loss"prior". It means how each practice would cost the project if we did not deal with by buying the control for that particular risk. To be noticed that cost for one ALE = the project budget /total of ALE, for example 100000/142=704.23 $, for each one of ALE. For example ALE for collective ownership practice =7.9,

then Loss "prior" = 7.9 x 704.23 $=5563$. Another for pair programming practice is 28.3, then Loss "prior" = 28.3 x 704 $=19930$.
The sixth column is priority. It means which practice must be dealt with firstly depending on the risk management budget available.
The seventh column is cost of control,, it is the cost needed to remove the risk for each practice. This cost of control for each practice can be obtained from invitation for tenders to get suitable control with suitable cost. Since currently we do not have enough data and according to the three categories of XP practices, we assume the following cost of control: the high practices take 3000$, the median practices take 2500$, and the low practices take 2000$ for cost of control.
The eighth column is deal, i means controlling t practice by buying the control which 100% removes the risk. The value 1 means the risk for the corresponding XP practice is dealt with and 0 not dealt with.
The ninth column is Loss "after". It means how each practice will cost the project if we deal with.
If we buy the control for the practice then the loss is 0$ otherwise the loss "after" = loss "prior".
The tenth column is cost of control needed. It means the total cost of all controls we buy to deal with all risks.
Table 4 shows general and software engineering risks for the risk tool elements: definition, example, duration, SLE, ARO and ALE.

4.4 Using the Proposed Tool

The following pseudo code illustrates the major steps of how to use the proposed risk management tool.

1. Define the Project budget.
2. Define the Risk management Budget %.
3. Enter the SLE value for each practice.
4. Enter the ARO value for each practice.
5. Tool calculates ALE for each practice , using this formula: ALE=SLE x ARO.
6. Tool calculates the cost pair ALE , using this formula: the project budget /Total of ALE.
7. Tool calculates the "Loss prior", using this formula: ALE x cost pair ALE.
8. Tool ranking the practices by Priority from the maximum "Loss prior" to the minimum "Loss prior".
9. Enter the cost of control for each practice.
10. Tool deals with each of the 12 XP practices as following:

```
for i=0 to 12
i=i+1
balance =risk budget - current cost of control
If balance = 0 then stop.
        else;
        take the high priority practice(i)
        if cost of control (i) < balance then take this
        control
         current cost of control =current cost of control
        +cost of control (i)
        else ;
        go to step 1
end
```

5. Results and Discussions

This section presents results obtained by using the tool and inserting some information including project budget, risk management budget, cost of controls, SLE, ARO into the tool to analyze the priority practices in order to decide which practice must deal with firstly.
The example shown (Figure 2) from the tool presents the project budget =100000$ and Risk Management budget = 15%=15000$. Because of the limitation of budget of risk management, the tool must deal with high priority practices which can cost the project a lot of money.

The priority practices that the tool has dealt with are: Pair programming, Sustained pace, Refactoring, Test-driven, development, Simple design and Small releases.

After buying and applying the control for each one, The cost the project decrease from 100000$ to 32746$ + the cost of control then the total loss is 47746$ rather than 100000$. The benefit now is 52254 $ (100000-47746=52254$), see Figure 2.

Table 5 shows the relationship between the Risk Budget % , amount , Risk control cost, Benefit and the covered XP practices for each steps. For example, if risk budget is 3% then risk budget amount will be $ 3000. In this case the tool deals with the first practice (Pair programming) that needs $ 3000 as cost of control. After buying and applying this control, the benefit will be $ 16930. For risk budget of 15% , the tool deals with 6 practices; Pair programming, Sustained pace, Refactoring, Test-driven ,development, Simple design and Small releases, for which the benefit is $ 52254. The user can increase or decrease the risk budget according to benefit he is looking for. A Return on Cost (ROC) may be defined as: ROC % = (Benefit /Risk control cost)*100.

6. Conclusion

This paper reviews various tools for Risk management in the area of software including Vanguard studio,

Pertmaster , @Risk, RiskEase, CrystalBall 7, RiskTrak [4, 6, 10, 13, 19]. The paper applies risk management on extreme programming practices. A tool is designed using MS-Excel 2007. This tool requires the definition of project budget and risk budget %. For each XP practice, the user enters the SLE ARO and the cost of control and the total cost of control needed (depending on risk management budget). The results found were presented in a previous section. Future work on the tool includes the following:

- SLE values: more studies are needed to get more realistic results according to local environment of usage.
- We suggest to get the cost of control for each practice by invitation for tenders.
- Dual interface language (English & Arabic): The tool may be enhanced by Arabic Interface for Arabic users.

Table 4: Comparison between general and software engineering risks elements

	Risk (General)	Risk (Software Engineering)
Definition	The possibility of something happening that will have an impact upon objectives.	The possibility of not applying the eXtreme Programming practices in project.
Example	Fire, Flood ,Virus, Hacker, Loss of data.	XP: changing requirements, cost of communication, difficulty in communication and incomplete requirements.
Duration	12 month period.	12 month / Project duration (months).
SLE	To estimate potential losses posed by threats. Single Loss Expectancy.	The weight of any XP Practice in the project (out of 100). Single Loss Expectancy.
ARO	The annualized rate of occurrence (ARO) is the value that represents the estimated frequency of a specific threat taking place within a one-year timeframe. The range can be from 0.0 (never) to 1.0 (at least once a year) to greater than one (several times a year). The probability of a flood taking place in Riyadh, KSA is once every 1000 years, the ARO value is 0.001.	The annualized rate of occurrence (ARO) is the range of not applied the practice of XP in a project within a one-year timeframe. The range can be from 0 (never) to 5(five a year). The probability of not applying the practice a collective ownership is once every project (if the project duration is 12 months) , the ARO value is (1x12)/12=1. And the probability of not applying the practice a continuous integration is once every project (if the project duration is 24 months) , the ARO value is (1x12) /24=0.5.
ALE	To derive the overall loss potential per threat . Calculate the annualized loss expectancy (ALE) per threat by using this formula ALE=SLE x ARO. So, if a flood taking place in Riyadh, KSA can cause $1000,000 in damages and the frequency, or ARO is 0.001 (indicating once in thousandth years), then the ALE value is $1000 ($1000,000 x 0.001 = $1000).	To derive the overall loss potential per practice of XP. Calculate the annualized loss expectancy (ALE) per practice of XP by using this formula ALE=SLE x ARO. So, if the practice a collective ownership is not applied in the project this can causes 7.93 in damages and the frequency, or ARO is 1 (if the project duration is 12 months), then the ALE value is 7.93 (7.93 x1 = 7.93). And if the practice a continuous integration not applied in the project this can cause 9.19 in damages and the frequency, or ARO is 1 (if the project duration is 24 months), then the ALE value is 4.6(9.19 x0.5 = 4.6).

Table 5: XP practices covered for various risk benefit.

Risk Budget		Risk control cost	Benefit $	Covered XP Practices
%	Amount $			
0	0	0	0	0
3	3000	3000	16930	1
5	5000	5000	26057	1, 12
8	8000	7500	34191	1, 12, 4
11	11000	10500	41261	1, 12, 4, 2
13	13000	13000	46648	1, 12, 4, 2, 6
15	15000	15000	52254	1, 12, 4, 2, 6, 7
17	17000	17000	56733	1, 12, 4, 2, 6, 7, 8
20	20000	20000	60071	1, 12, 4, 2, 6, 7, 8, 1
23	23000	22000	64268	1, 12, 4, 2, 6, 7, 8, 1, 9
25	25000	24500	67331	1, 12, 4, 2, 6, 7, 8, 1, 9, 5
28	28000	26500	70261	1, 12, 4, 2, 6, 7, 8, 1, 9, 5, 10
30	30000	28000	71500	1, 12, 4, 2, 6, 7, 8, 1, 9, 5, 10, 11

Acknowledgments

This work is partially funded by the Research Center of the College of Computers and Information Sciences, King Saud University.

References

[1] Assassa, G., Mathkour, H., Al Dossari, H. (2006),'Extreme programming: A case study in software engineering courses', Proceedings of the 1st National Information Technology Symposium, NITS, Riyadh, Saudi Arabia, pp. 233-240.

[2] Cameron, J. W. (2004), Managing Risk Across the Public Sector, The Victorian Auditor-General's Office, pp.1-8.

[3] Crystalball (2007), http://www.crystalball.com

[4] Kent Beck (2004), Extreme Programming Explained: Embrace Change, 2nd ed., Addison-Wesley, MA.

[5] Master Solutions Ltd. (2000), http://www.riskease.com/index.html

[6] Mathkour, Hassan, Aboalsamh, Hatim, Assassa, G., Al Dossari, H. (2006), 'Use of Extreme programming in software engineering Education: A pilot Study', The Engineering research Journal, Minoufia University, Vol. 31, No. 1, 2008, pp. 39-48.

[7] Misic, V.B. (2005), Perceptions of Extreme Programming: A Pilot Study. IEEE International Engineering Management Conference Proceedings, Volume 1, Issue Sept. 11-13, pp. 307-312.

[8] Paul Oldfield , Hüseyin Angay and Dan Rawsthorne, (2002), Risk To Pattern Table, Appropriate Process Movement, white paper, pp. 1-9. http://www.aptprocess.com/whitepapers/risk/ RiskToPatternTable.htm

[9] PertMaster Project analytics (2008), http://www.pertmaster.com

[10] Ramachandra, P., Kim, H.K., Kang, B., Ha, Y., Lee, R. (2006), Risk Management through Architecture Design, IEEE Fourth International Conference on Software Engineering Research, Management and Applications (SERA'06), pp. 386-395.

[11] Risk Management Policy Document (2005), Charnwood & North West Leicestershire PCT Risk Management Policy No. H&S/CNWL/1/5/8, 2005/2006, pp.1-42.

[12] RiskTrak, International (2007), http://www.risktrak.com/rstdemo.htm

[13] Shodan Input Metric Survey (2005), http://c2.com/cgi/wiki?ShodanInputMetric

[14] Sommerville, I. (2004), Software Engineering, 7th edn, Addison-Wesley, New York.

[15] Stewart Hayes BSA (2004), Software Tools for Risk Analysis and Management, pp.1-18.

[16] Sugianto, M. S. (2002), Risk Management In Project Management, The University Of Queensland Brisbane, Australia, pp. 32-61.

[17] Tsui, F. (2004), Managing Software Projects, Jones and Bartlett, London, pp. 107-122.

[18] Vanguard Software Corporation (2008), Management Systems for the Intelligent Enterprise. http://www.vanguardsw.com

[19] Wiegers, K. E. (1998), Know Your Enemy: Software Risk Management, Software Development magazine, October, pp. 1-6.

Risk Management Tool for Extreme Programming				
Project budget ?		100000	Cost per ALE = Project budget/Total Of ALE	704.23
Risk Budget % ?	15	15000		

PRACTICE No	XP PRACTICE	SLE ?	ARO ?	ALE ?	Loss "prior"	Priority	Cost Of Control	Deal	Loss "after"	Cost of Control Needed
1	Pair programming	8.09	3.5	28.3	19930	1	3000	1	0	3000
2	Test-driven development	9.51	1.5	14.3	10070	4	3000	1	0	3000
3	Continuous integration	8.97	1	9	6338	8	2500		6338	
4	Refactoring	9.02	1.67	15.1	10634	3	2500	1	0	2500
5	Collective ownership	7.93	1	7.9	5563	10	2500		5563	0
6	Simple design	9.57	1.17	11.2	7887	5	2500	1	0	2500
7	Small releases	8.15	1.33	10.8	7606	6	2000	1	0	2000
8	Planning game	6.94	1.33	9.2	6479	7	2000		6479	0
9	Customer On Site	8.75	1	8.8	6197	9	2000		6197	0
10	Metaphor	5.25	1.33	7	4930	11	2000		4930	0
11	Coding standards	9.19	0.5	4.6	3239	12	2000		3239	0
12	Sustained pace	8.64	1.83	15.8	11127	2	2000	1	0	2000
	Total	100		142	100000		28500		32746	15000

Cost of Control	
Category	Cost
H ?	3000
M ?	2500
L ?	2000

Final Results	
Loss prior	100000
Cost Of control	*15000*
Risk don't deal with	*32746*
Risk Management Total cost	47746
The profit	52254

Keys
SLE = Single Loss Expectancy.
ARO= Annualized Rate of Occurrence.
ALE = Annualized Loss Expectancy. ALE=SLE*ARO.
Priority : Which The practice must deal with firstly.
Deal : Controlling The practice by buying the control which removes the risk
Cost Of Control : The cost needed to remove the risk for each practice.
Loss "prior": How each practice cost project if we did not dealt with.
Loss "after": How each practice cost project if we dealt with.

Figure 2: XP risk management sheet