Requirement Risk Analysis in Software Development Projects

- An application of the DDP software tool

CS645 Software Requirements Specification and Analysis

Graduate Tutorial

University of Waterloo

December 19, 2007

Table of Contents

1. Introduction

1.1 Background

This tutorial focuses on the risk analysis process in the requirement phase of a software development project. The tutorial also introduces the risk analysis software tool; DDP (Defect Detection and Prevention) that NASA has developed. The tool is then applied to a case study of a Meeting Scheduling System.

Requirement Engineering risk management is the process of identifying, assessing and taking actions to reduce risks to an acceptable level. The process focuses on risks related to requirements and requirements processes which might lead to a delay, over-cost or even failure of the project. [1]

Risk analysis is a part of the risk management process. Risk analysis is the process of examining each identified risk issue to estimate the likelihood of a risk and predict the impact on the project. Before talking about risk analysis it is essential to define what a risk is and the content of a risk analysis, which the following section is about.

1.2 Definition of a Risk

A risk is a factor whose occurrence may result in some loss of satisfaction of some related objective of a project. A risk has a likelihood to occur and one or more consequences on the objectives that will experience unwanted events [1]

The two different risk types that correspond to the objectives are product-related risks, which have negative impacts on functional or non-functional objectives of the target product, and process-related risks which have negative impact on objectives of the development process. Some examples of product-related risks are failing to meet the product's specified requirements such as unsatisfactory security level, absence services and low quality of services. Some examples of process related risks are schedule delays, cost overruns, or outright cancellations. Risk is proportional to project size and inversely proportional to skill and technology levels. [1]

2 Risk Analysis in the Requirement Phase

Risk analysis involves consideration of the sources of risks, their positive and negative consequences on the project and the likelihood that these consequences may occur. In general the risk analysis process consists of three steps, which are listed below and will be briefly described in the following sections. [1]

- Identifying risks
- Assessing risks
- Control risks

2.1 Identifying Risks

The first step is to identify the potential risks in a project. A first step can be to break down requirements into categories and then analyze them group wise. Some examples of categories are; *must have*, *important to have*, and *nice but unnecessary to have*. The requirements can also be analyzed individually, so categorization is not mandatory. [3]

Risks may result from a variety of sources including the environmental interactions, the technology content, the implementation and operation approaches, the programmatic constraints and the project duration. [4]

Some typical identification techniques to use are interviewing key members of the project team, brainstorming with experts or/and stakeholders, checklists, fault trees, decomposition, component inspection, comparison with experience and examination of decision drivers. [1]

The identified risk factors can then be classified into different aspects such as stability, clarity and completeness. These risks have a significant impact on the overall software project process in terms of schedule, cost, scope and quality of the system to be developed. Some examples of categorized risk factors and their effect on the system are shown in Table 1. [2]

Table 1.Examples of classified risk factors and their effect on the system

Aspect	Risk factors	Effect on system
Stability	Requirements are not stable	Schedule, integration
Stability	External interfaces change	Design, testing, costs
Completeness	There are some requirements missing from common sense and engineers are not able to get those missing requirements into system	Quality, failure of system
Completeness	There are TBDs (To Be Defined) in the specification	Schedule, integration functionality
Completeness	Customers still have unwritten requirements/expectations	Failure of the system
Validity	Customers have a different understanding with regard to some requirements	Functionality
Clarity	Not able to understand the requirements as they are very ambiguous or as the explanation for the ambiguities is not satisfied for customers	Failure of the system.
Feasibility	Requirements are unfeasible from an analytical point of view such as technically infeasible or too expensive to implement	Schedule, costs, failure of the system.
Product	Some key functional requirements are hard to test	Failure of the system
Technical	The project size is large and no requirements management tool is used	Schedule, functionality, costs, failure of the system

It is generally accepted that poorly written, rapidly changing requirements are a source of project risks. Therefore, it is possible to identify risks by analyzing some attributes such as *ambiguity, completeness, understandability, volatility* and *traceability* of requirements. [5]

There are some metrics to identify risks according to these attributes. A metric for ambiguity is to count the weak phrases and optional phrases in the requirements document. Some weak phrases are *enough capacity*, *easy to use* and *normal.* This will indicate problems in the requirements document that can result in confusion and the need to take unplanned actions to resolve the questions. Thus, the higher the count of ambiguous terms, the higher the risk status of the project. [5]

A metric for completeness is to count not yet specified items in the requirements. To count the number of structure levels within the requirement document and also identify at what level the requirements are specified can be a metric for the attribute understandability. The traceability of the requirements upward to higher level documents and downward to lower levels such as code and tests can be measured as a percentage of trace up and down. [5]

Requirements volatility, which is defined as the rate of change of requirements, is always associated to be a source of project risks. A metric for volatility is to count the number of requirements changes in a given time period divided by the number of total requirements in the project. The total numbers of requirements is the sum of the number of imperatives and the number of items that follow continuances. Some examples of imperatives are *shall*, *must, will*, *responsible for* and *is required to* and some examples of continuances are *as follows, below* and *listed*. Though, note that the count for the number of changes is not by number of requirements changed, but by applying the same imperative/continuance count used to determine the base number of requirements. The requirements changing rate is normally more than 50% and classified as medium if at least 70% of requirements are stable and classified as low if 90% of the requirements are stable. In addition, the later in the life cycle changes are made to requirements, the more resources needed to implement them. Late requirement changes may also cause some kind of ripple effect, which causing additional changes in associated areas. The earlier in the life cycle the requirements stabilize, the lower the risk status of the project. [5]

In our case study of the Meeting Scheduling System, some identified risks are that the participants not check their email regularly and/or not reply on requests which may have a negative impact on the objective of scheduling convenient meeting dates. Another risk is that there could be difficult to find an available room with the required equipment for a particular meeting. If the system for some reasons responds late to the participants there is a risk that they will miss the notification before the meeting takes place. Also, there is a risk that important meeting participants can have a last minute obstacle which the system does not know. Based on the list of risks next step is to determine how serious each risk is. [1]

2.2 Assessing Risks

The third step is to assessing both the risk and the consequence(s) a likelihood to occur, respectively. Also, severities of the consequence need to be assessed. There are two types of assessment; it can either be qualitative or quantitative. In qualitative assessment quality scales like *very unlikely* to *very likely* can be used. An example is that risk 1 is *likely* to occur and risk 2 is *possible* to occur. The severity of consequences can also be assigning values on a scale from

low to *catastrophic*. Qualitative assessment uses numerical scales. In this case, the probability of a risk to occur can for example be 0.7 and the value for the severity of the consequence can be 8 on a scale from 1 to 10. [1]

Based on the risk assessments the next step identifies and analysis possible countermeasures that can be done to reduce them. The last step in the risk analysis process is control risks by mitigate them.

2.3 Control Risks

Risk control is about planning and taking steps in a cost-effective way to mitigate risks in a project. These steps are the countermeasures to reduce risks, especially the high ones. These countermeasures will modify the existing requirements and also yield new ones. Typical techniques to identify countermeasures in a project are interviewing stakeholders, cost-models, risk reduction strategies such as reduce risk or/and consequence likelihood, avoid risks or consequence of them and mitigate risk consequence. Another alternative is to instantiate generic measurements to the specific context of the project using Boehm's list of top ten risks which also identifies some countermeasures. [1]

When the countermeasures are identified an optimal subset of them needs to be selected. This selection depends on the cost-effectiveness of countermeasures and their contribution to other non-functional requirements that includes cost limitation requirements. [1]

3. Types of Risk Analysis Methods

3.1 Methods in General

There are many names and variants of risk analysis methods. Also, several of the different risk analysis techniques are branch specific. Every type has specific definitions, structures, calculation models and different ways to present the final result. However, in general the methods can be categorized as quantitative and qualitative which are briefly described in the following sections. Some typical risk analysis techniques are network analysis, decision trees, cost models and performance models. [2] A software tool for Failure Mode Risk Management developed by NASA, called DDP, is described in detail in Section 4.

3.1.1 Quantitative Risk Analysis

Quantitative risk analysis, which is based on quantitative assessments of risks, is fairly limitedly used. The method uses two elements: probability and likely loss. By multiplying the probability and the likely loss an ALE (Annual Loss Expectancy) is produced. This ALE can be used to rank events in order of risk and to make decisions based upon this. There are several drawbacks with qualitative risk analysis such as the lack of an accurate probability database, the fact that the probability used is usually unique to the case, and that the expected loss is hard to establish. [6]

3.1.2 Qualitative Risk Analysis

Qualitative risk analysis, which is based on qualitative assessments of risks, is widely used and can be used in most projects to determine which risks are important enough to manage. The method is useful when reliable data is not available because no probability database is required. The quality risk analysis has the purpose to evaluate each risk and designate each risk as *high*, *medium*, or *low*. This rating depends on two criteria; the severity of impact and the probability of the event occurring, which was mentioned earlier in Section 2.2. [6]

4 DDP - Defect Detection and Prevention

4.1 Introduction and Overview of DDP

NASA has developed a software tool, called DDP that has the goal to find mitigations that cost-effectively reduce risks and give aid in the achievement of continuous risk management. The process can be used for different purposes. The most common purpose is to use the tool to reduce risks. Another purpose is to plan how to accept more risk in exchange for benefits such as reduced cost and better schedule and more functionality in a project with currently few risks. DDP can also be used to maintain a desired risk profile through the lifetime of the project. Finally, it can be used to assess an unknown risk status in a project. [6]

The process makes the user think about three important key words; requirements, aka *objectives*, potential risks, aka *failure modes*, and what can be done to prevent or detect failures, aka *PACTs (Preventative measures, Analyses, process Controls and Tests)*. The process analyzes the consequences of the potential risks by scoring their impact on the requirements if they should occur. This result is a list of requirement-driven risks where failure modes are derived their criticality from their impact on possibly, weighted requirements. The process also scores the effect which is the proportion by which risk is reduced if any mitigation is applied. The failure modes may be weighted by a likelihood of occurrence even if nothing is done. The result of DDP is an optimized collection of mitigation activities performed on a project. [4]

Through objectives, failure mode and PACT tree evolution; the process utilizes all available information to provide the most up-to-date view of the risk landscape in a particular project. By combining both intuitive and analytical information, the DDP process can be implemented over the entire project life cycle. Thus, the process is a systematic top-down approach which integrates bottom-up information. [4]

The process consists of the following three steps:

1) Develop the Requirements Matrix
2) Develop the Effectiveness Matrix
3) Optimize the Residual Risk

The key parts are the project objectives, the failure modes and the PACTs which occur at various levels that range from high mission level all way down to the low device level. The DDP process is adapted to evolve with the project development cycle to allow risk elements to be identified as early as possible. This also allows the risks to remain consistent with the necessary initial allocation of resources and facility scheduling. [4]

There are various techniques to find the failure modes, some of them are mentioned in Section 2.1. As the lower levels of failure modes are identified, they will eventually reach the "root cause" of the risks. One of the main goals of the DDP process is to allow decisions that are "good enough" to be made at as high a level as possible. The identified failure modes are critical to the DDP process and show up in both the Requirements Matrix and in the Effectiveness Matrix which are both described in the following sections. [4]

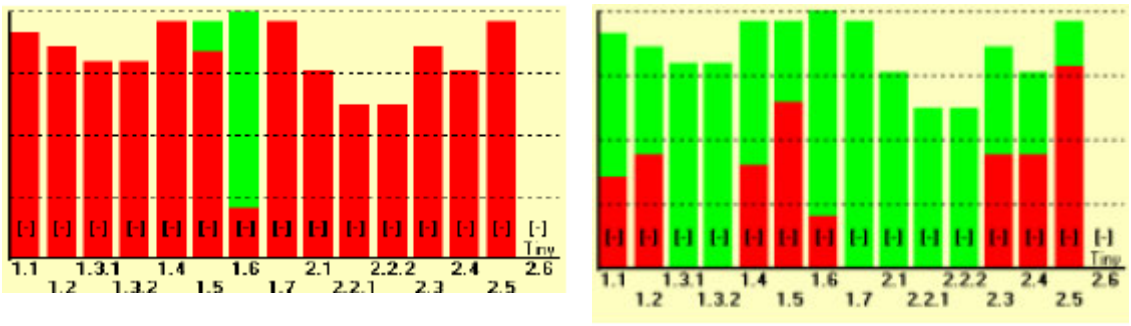

Figure 1. Typical outputs from DDP in form of bar charts. Red parts means remaining risks and green parts means mitigated risks, which are associated with a cost.

4.1.1 Step 1: Developing the Requirements Matrix (RM)

The goal of the first step in the DDP process is to develop a prioritized set of potential failure modes. Another part of the goal may also be to identify which requirements are producing the most risk, aka *risk-driving*, and which requirements are less relevant. [4]

To build the Requirement Matrix, begin with a list of the project's specified requirements. These may be individually weighted, relative to how important they are. Identify and make a list of the failure modes. These may be weighted by their probability of occurrence when no mitigation activities are applied. In most cases this weight factor is chosen to be 1.0. [4]

The requirements and failure modes can now be used to build a Requirement Matrix, see Figure 2, that for each pair specifies an estimated loss if the risk occurs. This value is called *impact*. High-quality impact values are obtained by domain experts, estimates, or models and simulations. After all these steps have been taken the matrix is filled by all the impact values. It is now possible to calculate the risk criticality of each failure mode by using Equation 1. Summing down the columns, weighted by the relative importance of each requirement and the likelihood of each failure mode yields the criticality of each failure mode. It is important to note that the risk criticality can be modified by changing the requirements or their relative importance. [4]

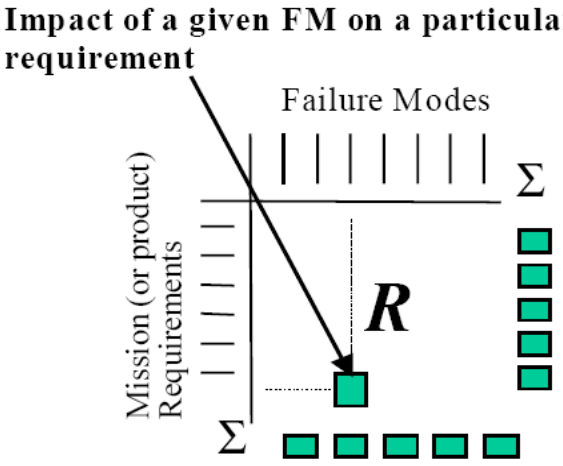

Figure 2. The Requirement Matrix maps the impacts [%] of each failure mode on each requirement

Equation 1. The formula calculates the criticality for a failure mode

$$Criticality(FM) = Likelihood(FM) * \sum_{req} (\mathrm{Im}\, pact(FM, req) * Weight(req))$$

Next step is to sort the failure modes by the criticality and find the so called *tall pole*, which is the one with highest criticality value. The DDP will now provide a bar chart visualizing the critical impact, by decreasing order, of the various risks on all the listed requirements. [4]

Now, calculate the loss of objective, which is in percentages, for each requirement by using Equation 2. This equation is a summation across the rows and yields the extent to which each requirement is at risk. These calculated values make it easy to identify the most *risk-driving* and also the irrelevant requirements in the project. The risk-driving requirements are valuable information and they may then be re-examined for their necessity. [4]

Equation 2. The formula calculates the loss of objective for a requirement

$$Loss\ (req) = Weight\ (req) * \sum_{FM} (\mathrm{Im}\ pact\ (FM\ , req) * Likelihood\ \ (FM\ ))$$

The final result of the first step in the DDP process is the Requirement Matrix. The tool will also provide a bar chart visualizing the critical impact, by decreasing order, of the various risks on all listed requirements. [4]

To apply this first step in the DDP process on the Meeting Scheduling System the requirements and the potential risks have to be identified. Some of the requirements for the system are defined from the description of the case study. [1] The risks, used in this example, are identified in section 2.1. The values of the weights of objectives, likelihood of risks and the impacts are made up to have some numbers for this example. The result is shown in the Requirement Matrix in Figure 3.

RM for Meeting Scheduling System

"Tall poles"

		Risks (Failure Modes)					
Objectives (Requirements)	**Weight (obj)**	Participant does not read emails	Participant does not reply on requests	Room with required equipment is not avaliable	System response is too close to meeting	Important participant has last-minute change	**Loss of objective**
Likelihood(r)		0,4	0,3	0,1	0,3	0,5	
Time taken to schedule meetings reduced	0,5	0,6	0,8	0,2	0,7	0,2	0,405
Send out notification asap when time and place are found	0,4	0	0,8	0	1	0,2	0,256
Participant avarage attendance increased	0,3	0,8	0,8	0	0,8	0,5	0,315
Schedule conflicts reduced	0,6	0,2	1	0	0	0,7	0,438
Risk criticality		0,264	0,468	0,01	0,297	0,375	

"Risk-driving requirement"

$$Criticality(FM) = Likelihood(FM) * \sum_{req} (\mathrm{Im}\ pact(FM, req) * Weight(req))$$

$$Loss(req) = Weight(req) * \sum_{FM} (\mathrm{Im}\ pact(FM, req) * Likelihood(FM))$$

Figure 3. Requirement Matrix for the Meeting Scheduling System

4.1.2 Step 2 - Developing the Effectiveness Matrix (EM)

The goal of the second step is to develop a set of options, which are a combination of PACTs, for preventing or detecting the failure modes. Begin to list the countermeasures, the identified PACTs. Then build the frame of the effectiveness matrix by adding these PACTs and the failure modes used in the requirement matrix. The PACT options for detecting and preventing failure modes range from design rules to system level tests and they have varying effectiveness against different failure modes. [4]

There are different ways to determine the effectiveness of a given PACT on a given failure mode. This effectiveness can be determined by experts, historical data or results from experiments or measurements. Then, build the Effectiveness Matrix (EM), see Figure 4, based on these effectiveness values. This matrix captures the effectiveness of each PACT against each failure mode and is scored as the fractional reduction in the likelihood of occurrence of the respective failure mode. The next step is to calculate the combined risk reduction, also known as the *Residual Risk*, for each failure mode by using Equation 3. [4]

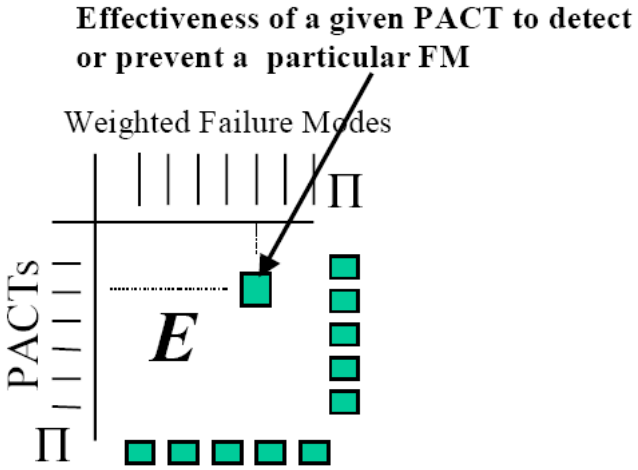

Figure 4. The Effectiveness matrix maps the probability of detecting or preventing each failure mode by each PACT should the PACT be implemented. (e.g. % of chance of failing to detect or prevent)

Equation 3. Formula to calculate the combined reduction of risk for each failure mode

$$combined\ \mathrm{Re}duction(FM) = 1 - \prod_{PACT}(1 - \mathrm{Re}duction(PACT, FM))$$

Then calculate the overall single effect of each PACT by using equation 4. [16]

Equation 4. The formula calculates the overall single effect of each PACT

$$overallEffect(PACT) = \sum_{FM}(\mathrm{Re}duction(PACT, FM) * Criticality(FM))$$

These calculated values are then used to identify the most globally effective PACT. Note that the formula in Equation 4 is a simplification for calculations by hand. The DDP tool calculates a more refined option for overall effect of a PACT. This option is based on failure modes with their likelihoods as already reduced by whichever of the other PACTs that have already been selected. [4]

The result of the second step in the DDP process is the Effectiveness Matrix, see Figure 4, and a risk balance chart, see Figure 5. This chart shows the residual impact of each risk on all objectives if the corresponding countermeasure is selected. The residual risk is the *expected value* of the failure mode that means the extent of its impact times how likely it will occur. [4]

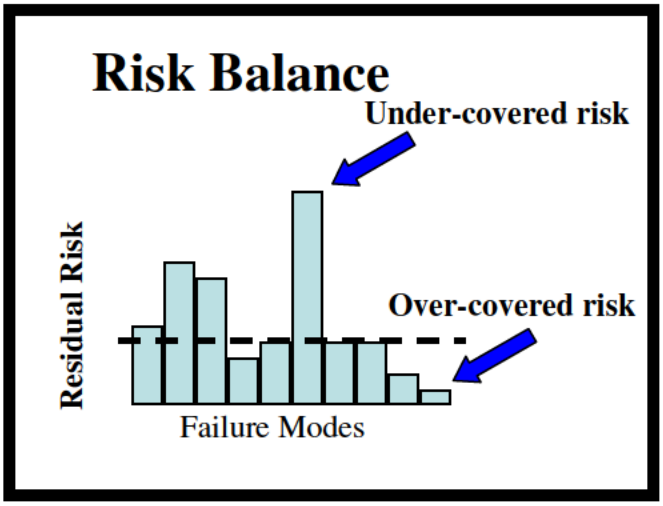

Figure 5. Risk balance bar chart output of step 2

The DDP also prepares the team with a collection of possible PACTs and the costs associated with them. Sometimes the project prior to the evaluation has already chosen a particular baseline for the countermeasures to be selected. This means that there are some additional constraints on the PACT selection. [4]

To apply this second DDP step on the example with the Meeting Scheduling System some possible countermeasures are defined. The identified risks are the same as used in the Requirement Matrix in Figure 3. The effectiveness values filled in the matrix are made up, without expert knowledge, for this example. The result is shown in the Effectiveness Matrix in Figure 6. The most and the less global effective PACTS can easily be identified by calculating the overall single effect of countermeasures and compare the values. The less and the most reduced risk is also identified and marked in Figure 6. [4]

EM for Meeting Scheduling System

Less global effective PACT

	Risks (Failure Modes)					
Countermeasures (PACTs)	Participant does not read emails	Participant does not reply on requests	Room with required equipment is not available	System response is too close to meeting	Important participant has last-minute change	**Overall single effect of countermeasure**
Likelihood(r)	0,4	0,3	0,1	0,3	0,5	
Email reminder sent	0,7	0,7	0	0,1	0	0,5421
Change the meeting, greater time range	0,2	0,2	0	0,1	0	0,1761
System has access to personal e-agenda	0,3	0,2	0,1	0,2	0,3	0,3457
Change the meeting, less constraints (equipment)	0	0	0,9	0	0	0,009
Cancel a meeting and send email confirmation	0,8	0,8	1	0,7	0,9	1,141
Combined risk reduction	0,9664	0,9616	1	0,8056	0,93	

Most global effective PACT

"Most reduced risk"

$$combined\ \text{Re}duction(FM) = 1 - \prod_{PACT} (1 - \text{Re}duction(PACT, FM))$$

$$overallEffect(PACT) = \sum_{FM} (\text{Re}duction(PACT, FM) * Criticality(FM))$$

Figure 6. Effectiveness Matrix for the Meeting Scheduling System

4.1.3 Step 3- Optimizing the Residual Risk

The third step in the DDP process has the goal to select the optimal, or near optimal, combinations of PACTs which minimize the risks subject to various project resource constraints such as time and money. There is resource costs associated with each PACT. These costs need to be estimated by experts. Since each PACT has varying effectiveness against different failure modes, it is possibly and necessary to "mix and match" different PACT combinations each of which has associated resource costs. Various effectiveness against different failure modes are for example a design decision, such as additional shielding or redundancy, may require more mass, while a decision to perform a test means allocation of schedule and money. As a result of this step DDP provides a risk balance chart which displays the status of each of the failure modes.

Also, it taking into accounts the mitigating effects of the selected PACTs. The risk balance chart from step 2, see Figure 5, can be used to identify the under-covered risks and the over-covered risks in the project. The under-covered risks, the tall poles, are the most important to address and assign resources to. The resources addressed to the over-covered risks can be moved somewhere else to make the most of the projects resources. The line in the risk balance bar chart, in Figure 5 and 7, is the location where each failure mode is receiving risk reduction proportional to its importance. [4]

The result of the third step in the DDP process is an optimal, or near optimal, selections of countermeasures, PACTs, that the DDP has found by using an optimization method called simulated annealing. This method is used because it is simple to implement and the convergence is fast. [4] The optimal risk balance bar chart is shown in Figure 7.

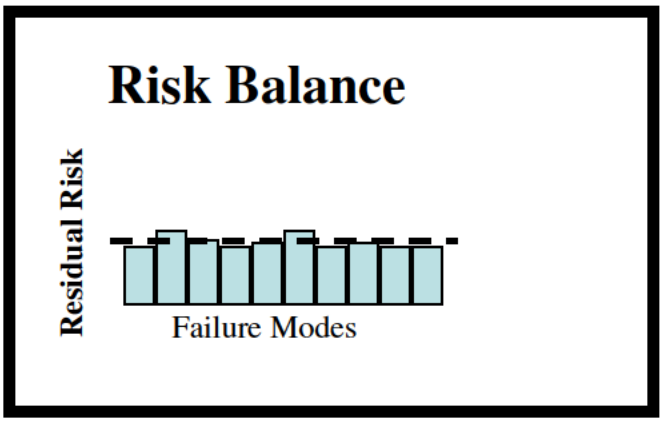

Figure 7. Risk Balance bar chart when optimized

The third step in DDP process is not applied on the Meeting Scheduling System because the requirements, risks and countermeasures used in previous examples are just small subsets of the project.

4.2 Advantages and Drawbacks of DDP

DDP has many advantages and has opened doors for NASA and other software development companies. Some of them are listed below.

- Provides a good illustration of the kind of technology supporting the risk analysis process both in the requirement evaluations and the entire project life cycle [1]
- Allows capturing changes in the requirements during the project life cycle. For example when the requirements and hardware maturity evolve and their refinement to lower levels may result in modification of various levels of requirements. Similarly, when the evolution of the design results in evolved failure modes. The DDP process captures these

changes, evaluate their impact on the “tall pole” failure modes, identify risk-driving requirements, update the Effectiveness matrix and re-evaluate the Residual Risk [4]

- Links explicitly to objectives and requirements [1]
- Reduces development and operating costs [4]
- Reduces risk [4]
- Increases safety and reliability [4]
- Designs better software architecture and more maintainable systems [4]
- Enable companies to handle more complex systems in the future [4]

There are also drawbacks by using the DDP tool. Some of them are listed below.

- Takes lots of time and effort to gather the information DDP requires as input. In comparison the identification of objectives and detailed assessment of individual mitigations are not usual requirements in early-lifecycle risk management techniques. [7]
- Shortcomings remain in the DDP model. The tool lacks of the ability to deal with distributions and uncertainty, especially with complex models of utility and also with the fault-tree gates and event-sequence diagram.[7]
- The lack of any such automated capability to select optimal combinations of PACTs. This forces human users to manually pick their suite of risk mitigations, guided by various graphical presentations of information. Manual selection takes time, and may well fail to find anything near an optimal solution. [8]
- The combination rules that are used to find the optimal combination of PACTs are complicated and require explanation [8]
- Scepticism of validity of results, based as they are on simplistic model and multitude of estimates [8]
- The data and the estimates are particularly weak for software [8]

5. Summary and Conclusions

The risk analysis process in the requirement phase in a software development project consists of three steps: identifying, assessing and controlling risks. These steps systematically identify the lists the possible risks, assess them, qualitatively or quantitatively, based on how serious they are and control them by making plans and taking actions to mitigate them.

Not every risk is fully controllable, and several risks exceed the authority of software managers. Nonetheless, risk analysis and assessment methods are quite effective in the identification of significant problems. Once problems are identified and examined, solutions can often be developed.

The DDP approach has many advantages such as provides a good illustration of the kind of technology supporting the risk analysis cycle during requirements evaluation. The approach can be used during the whole project life cycle but in this tutorial it is described how to use in the requirement phase. DDP covers most of the risk management concepts such as identifying, assess and control risks. It also links explicitly to objectives and requirements and exhibits typical quantitative reasoning schemes that are available for requirements evaluation. DDP has tool support for carrying out such reasoning and for visualizing results.

References

[1] A. van Lamsweerde (2001). *Requirements Engineering – From System Goals to UML Models to Sopftware Specifications*. Wiley 2007. Course Package CS 445 Fall 2007

[2] L. Jiang A (2005) [Thesis] *Framework for the Requirements Engineering Process Development*, Department of Electrical and Computer Engineering, Calgary, Alberta. Retrieved November 24, 2007, from Web site: http://www.enel.ucalgary.ca/People/eberlein/theses/LiJiang.pdf

[3] G. McGraw (2006). Software Security: Building Security In. 1st Edition. ISBN-10: 0-321-35670-5

[4] S. Cornford et al (2001). *DDP – A tool for life-cycle risk management.* Jet Propulsion Laboratory, California Institute of Technology, California. Retrieved November 21, 2007, from Web site: http://ddptool.jpl.nasa.gov/docs/f344d-slc.pdf

[5] A. Buttigieg. *Risk Management in a Software Development Life Cycle.* Retrieved November 29, 2007, from Web site: http://www.cis.um.edu.mt/~abut/

[6] NASA (2007). *DDP Tutorial on the Web.* Jet Propulsion Laboratory, Californian Institute of Technology, California. Retrieved November 29, 2007, from Web site: http://ddptool.jpl.nasa.gov/

[7] M. Feather et al (2001), *Got Risk? A Risk-Centric Perspective for Spacecraft Technology Decision Making.* Jet Propulsion Laboratory, California Institute of Technology, California. Retrieved November 27, 2007, from Web site: http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/38292/1/03-3235.pdf

[8] S. Cornford et al (2002). A Risk Centric Model for Value Maximization. California Institute of Technology, California. Retrieved November 30, 2007, from Web site: http://sunset.usc.edu/events/2002/cocomo17/agenda_cocomo17.html