Volume 8, No. 5, May-June 2017
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

# A Vague Rule Based Analysis Model for Software Risk and Quality Evaluation

Kavita Dahiya
Research Scholar
Computer Science and Application Department
MDU, Rohtak

Dr. Priti
Assistant Professor
Computer Science and Application Department
MDU, Rohtak

***Abstract:*** A software system is defined large number of modules and integrated components. While developing a software system, it is required to analyze the risk factor or the expected quality at the planning stage. At this stage, the requirement and expectation based mapping can be obtained to identify the risk and the quality. These measures directly represent the chances of system failure, cost of project, delay of project etc. In this research, multiple aspect based software quality estimation is provided. These multiple aspects include the entity specific, procedural, resource specific and the object specific. The paper has defined a vague formulated risk evaluation model with multiple associated stages. At the earlier stage, the risk categorization is provided. Later on, the vague based individual aspect risk and the category specific aggregative risk evaluation is provided. In this paper, the risk estimation measure is applied on Cocomo NASA 2 projects datasets. The result shows that the model provided effective evaluation of software projects.

***Keywords:*** Risk Quality, Vague Rule, Software Metrics, Process Metrics

## INTRODUCTION

The quality of software system affects from various errors, faults and associated risks. Various software metrics are available to analyze the software system at each process stage of S DLC ( Software D evelopment L ife C ycle). The metrics are defined to estimate the software system based on different p arameters including the p erformance, r eliability, cost, r eusability, et c. These p arameters c an be a dapted individually or i n c ombined f orm t o t ake t he de cision regarding the effectiveness of the software system. The functionality d riven m ethods ar e av ailable t o an alyze t he quality o f s oftware s ystems. A t th e in itial l evel, th e s tatic analysis is applied on software system to take the outer view. In this s tudy, th e s oftware s ize, tim e and cost d riven parameters ar e co nsidered. The u sed m anpower, development schedule can be considered to analyze the complexity o f the s oftware s ystem. L ater on t he d ynamic parameters are applied to evaluate the quality and reliability of software systems. The run time observation with different test methods can be applied. The analysis includes the deep white box testing methods as well as the run time evaluation using b lack b ox t esting. These dynamic m ethods can estimate th e s oftware cost, q uality, e fforts, reusability requirement, m aintenance r equirement, e tc. In this s ection various kinds of available software metrics and the associated metrics categories are discussed. The section has given a c lear vi ew t o the va rious a daptive m ethods a nd metrics available to analyze the software system.

In the wider form, the software metrics defined to analyze the s oftware s ystem ar e d ivided into three m ain cat egories shown in figure 1. These metrics can be applied in composite form at different stages of SDLC to perform the evaluation at each l evel. Each of the c ategories also having a n umber of inclusive parameters, c onstraints a nd m ethods. In c omposite form, t hese m etrics ar e a ble to ch aracterize t he software system quality in the adaptive way. The product level, process l evel and t he required r esource level evaluation are required t o o bserve e ach i nclusive aspect of s oftware systems.

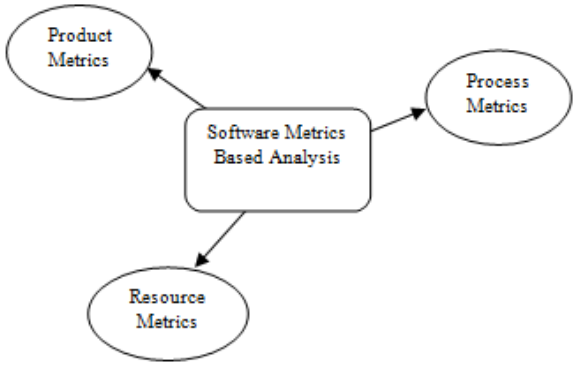


Figure 1. **: Methods of Software System Evaluation**

### *1.1 Product type metrics:*

These are the static measures used to analyse the software size, c omplexity, testability m easures and th e e xistence of common bugs in software systems. The main effect of these metrics is o n t he p erformance o f software systems. T hese metrics a lso a ble to ge nerate t he evaluation regarding t he portability, efficiency and the cost. The time line of software development, required software efforts or manpower can be estimated based o n the size l evel es timation. T he structure and the c omponent d ensity b ased ob servations a s the measure the quality of the software system.

### *1.2 Process type metrics:*

The p rocess metrics an alyze the software d evelopment activities a nd th e c ommunication f low within th e s oftware project. The methods, measures and standards are defined to analyze the control behavior and communication flow within the pr oject. T he pr oductivity, effort, c ost and timeline measurement can be obtained from the process metrics. The software p roject scheduling w ith milestone setup can b e identified f rom t hese m etrics. The m aintenance or r euse of software p roject is a lso directed from these metrics. T he process driven faults an d the complexity challenges can be identified from process based metrics.

### *1.3 Resource metrics:*

The resource metrics are able to identify the requirement of software systems at various stages, including the development stage, design stage and the maintenance stage. The resource metrics are having effective contribution in planning of software system and design. The human and non-human resource estimation, the task assignment can be regulated based on these metrics. The software cost and the schedule can be predicted based on the resource metrics. The metrics are able to generate the quantified decision on requirement and availability measures. The component specific and the aggregative decision can be taken based on these metrics.

In this paper, an analytical observation on various aspects of software quality evaluation is considered with mathematical formulation. The paper has analyzed the software project under product level, resource level, software release level software aspect estimation. Each of the aspect is analyzed individually under vague rule and later on the composite features are evaluated. In this section, the brief introduction to the software system evaluation and metric based estimation is provided. Various categories of metrics are discussed in this section. In section II, the work provided by earlier researchers is discussed. In section III, the proposed research methodology with algorithmic method is discussed. In section IV, the results obtained from the algorithmic implementation are discussed. In section V, the conclusion obtained from work is presented.

## RELATED WORK

Lot of work is earlier provided by different researchers for software quality, cost and risk estimation. The parameter specific and the operation specific work is provided by the researchers to evaluate the software quality. Reham Ejaz et. al.[1] has defined a quality assurance model to analyze the defect in software system. The software development effort and criticality estimation is provided to perform the risk assessment and verification at the analytical stage. Author estimated the quality driven evaluation identify the component and the software cost. Philip Koopman et. al.[2] has evaluate the risk factors for the embedded software system. The software development problems are categorized by the author with each of the development stage. The system evaluation, categorization and the product evaluation was provided to measure the issue vectors. A work on software safety and process improvement for different real time projects was identified by Victor R. Basili et. al.[3]. Author identified the software safety and generated the hardware specific evaluation to identify the associated deficiencies. The process level quantization and the risk evaluation were provided to observe the software safety at product and process level. The design phase evaluation and safety risk estimation was provided by the author. A detailed industry based study work on risk for software change was identified by Shihab et. al.[4]. In this study more than 450 developers of large enterprises was involved and identify the software level changes at code and design time. The design review and testing was provided by the author to determine the risk evaluation. The additional attention at design time was provided to formulate the associated changes more accurately. The modification made by the developer and its impact at different phases of software development was also identified.

G. Antoniol et. al.[5] has identified the requirement of software evaluation and research with specification of new trends and needs. The opportunistic challenges and the key factor evaluation for software system was also formulated by the author. The benchmark requirement of software development and its integration in real environment was identified by the author. Tomaszewski et. al.[6] has applied a comparative evaluation on software fault and change identification for the industrial projects. The fault feature identification for different categorize of faults and risk was evaluated by the author. The system risk and the fault identification with software change was formulated by the author. Islam et. al.[7] has defined a risk management and evaluation model for software development project. Author identified the unanticipated problem with pose and potential risk evaluation. The stage driven development and component integration was provided by the author. The technical development and the risk evaluation was provided by the author to estimate the software quality. Durisic et. al.[8] has defined a work on product quality estimation for software change estimation. The metric and complexity specific risk evaluation was provided to identify the software cost. The phase specific evaluation and the architectural change were identified by the author to estimate the quality for different functional aspects. The quality risk evaluation and the metrics level support was identified by the author. Matsuo et. al.[9] has defined the task diversity analysis for software system development. A contract model for risk aversion was provided by the author to estimate the situational risk. The distributed task model is defined to reduce the time and cost of development for larger projects. The situational estimation was also provided to improve the software performance. Islam et. al.[10] has identified the security risk with involvement of human factor and management. The risk identification, analysis and mitigation was provided by the author.

Some other work analytical, study and the method evaluation was provided by the researchers to explore the software system artifacts. Li et. al.[11] identified the systematic and the criteria driven evaluation to generate the priority specific reviews. The process specific evaluation and the prioritization was provided by the author to generate the quantization of the software artifacts. The product specific risk, dependency and the cost evaluation for verification and validation was provided by the author. Witmer et. al.[12] has mitigate the risk based on the requirement preservation for the multimedia software system. The requirement observation for different media forms including image, textual data and videos was provided by the author. The requirement and the design time improvements were also identified to improve the software system validity. Idongesit et. al.[13] has performed the assessment of software risk and evaluate the associated efforts. The quantitative and intuitive process analyses with inherent risks were explored by the researcher. A combined quantitative risk model was defined to reduce the cost and improve the software quality. Dhlamini et. al.[14] has defined an intelligent risk management tool to reduce the software development and provided the risk assessment at lower level. Author also defined the risk management and risk evaluation to investigate the development aspects at lower level. The

intelligent aspect m odeling a nd th e im proved s oftware system d evelopment w as a lso provided by t he author. Layman et. al.[15] has defined a work on component driven analysis o f development, s cheduling a nd t he s oftware delivery. The artifact driven estimation and the trend based measurement were also provided by the author.

From th is s ection, it is identified th at t o im prove th e software system development, a prior analysis and evaluation stage is required to identify the future risks and to improve the s oftware q uality. I n t his p resent work, a vagueg r ule based software risk and quality estimation is provided under different integrated aspects. In next sections, the model and the associated experimentation is also described.

## VAGUE RULE ADAPTIVE SOFTWARE QUALITY EVALUATION MODEL

In t his present w ork, a m ore ef fective and rule based method is p rovided t o e stimate th e s oftware quality. The quality of the complete system is not based on one factor but it includes more 10 different factors. These factors are representing t he quality o f s oftware s ystem und er different perspectives in cluding th e e ntity s pecific, o bject s pecific, process specific o r t he resource specific. T hese al l q uality vectors are he re estimated individually and i n combination. In c ombination, the q uality is d efined respective t o s ome quality or the risk category. Such as the programmer ability, OS s upport, technology i s t he i ndividual quality o r risk factors which are combined to represent the platform risk. In same way, s ix di fferent c ategories of s oftware q uality measures u nder different as pects a re g enerated. A nd e ach category is d efined by m ultiple in dividual quality o r risk factors. To evaluate these risk factors in composite form, a vague rule based method is applied. Vague defines the mathematical r ules that c ombines t he at tributes based on union, intersection or s ome ot her a ssociation rules. F inally, the quality o f th e individual module or software system is evaluated. The work flow of the proposed mathematical rule based software quality evaluation method is shown in figure 2.

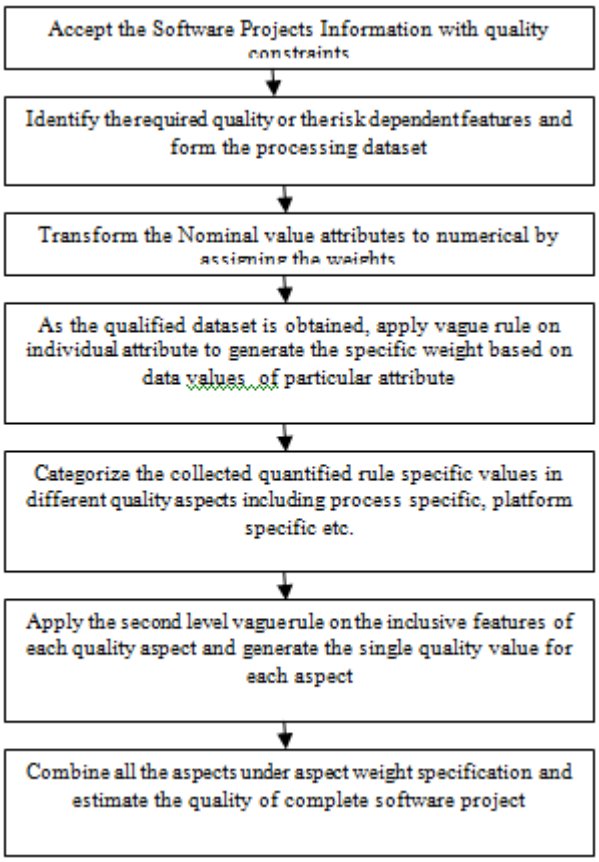


Figure 2. **: Flow of Work**

Here figure 2 has described the complete work to analyze the software system under different aspects and by apply the value specific mathematical ev aluation. I n t his method, a t first th e in dividual a ttribute is a nalyzed und er va gue r ule formulation a nd l ater on th e aspect in clusive a ttributes are analyzed c ollectively. Based o n t he r equirement an d t he observation, t he w eights are assigned to each aspect as well as each quality feature.

The p resented w ork begins, as th e s tatistics of s oftware project g et a vailable. This d ataset is h aving th e descriptive features as well as the q uality specific features. To process the p roposed algorithmic a pproach, th e quality s pecific features are s eparated and presented as t he featured dataset. The raw dataset c ollected is having the no minal values. To perform the value specific rule formulation, it was required to t ransform t he dataset i n numerical f orm. F or t his, t he mapping of each nominal value to specific numerical value is done. T his mapping is ba sed o n the weight o f t he relative nominal va lues such a s vh (very hi gh) i s a ssigned by hi gher numerical va lue a nd vl (very l ow) i s a ssigned by l east numerical v alue. A s t he n umerical dataset i s obtained, t he next w ork d efined he re is t o apply t he va gue r ule on e ach individual a ttribute to a ssign t he w eights ba sed on v alue observation. A fter t his s tage, v ague t rained dataset i s obtained with relative numerical weights.

In second stage of this model, the static analysis on the quality f eatures of software projects is d one t o i dentify th e relational aspect. A t this s tage, t he aspect s pecific f eatures are identified and defined as the composite feature set. In this work, 6 such software quality aspects are identified including the p roduct quality, p rocess q uality, p latform q uality, personal quality, Reuse quality and Schedule Quality. Each of t he a spect is f urther ha ving multiple i nclusive f eatures.

The mathematical rule framed algorithmic method is shown below.

**Algorithm 1**

```
VagueRuledAnalysis(projects)
/*projects is the list of software projects defined with relative characteristics*/
{
1.  for i=1 to projects.length
    [Process the projects]
    {
2.  Pfeatures=projects(i).GetFeatures()
    [Obtain all the project features]
3.  For j=1 to prfeatures.length
    [Process all the features with nominal values]
    {
4.  tfeatures(j)=ApplyRule(prfeatures(j))
    [Apply first level rule to transform the nominal values to numerical features]
    }
5.  ScheduleRisk=GetVagueScheduleRisk(tfeatures)
    [Identify the schedule risk from the generated features]
6.  [SoftRelRisk DBSizeRisk ProcessComplexityRisk]=GetVagueProductRisk(tfeatures)
    [Obtain the Vague adaptive product risk]
7.  [TimeBoundRisk MemoryConsRisk DBSizeRisk MachineVolRisk]=GetVaguePlatformRisk(tfeatures)
    [Identify the VAgue based Platform Risk]
8.  [AnaCapRisk AppExpRisk LangExpRisk ProCapaRisk MacExpRisk ModProgPracRisk]=GetVaguePersonalRisk(tfeatures)
    [Identify the vague based Personal Risk]
9.  [ToolUseRisk TimeConsRisk TurnAroundTimeRisk ProgCapRisk MachExpRisk]=GetVagueProcessRisk(tfeatures)
    [Identify the Vague based Process Risk]
10. ScheduleRisk=VagueAgg(ScheduleRisk)
    [Apply Vague Aggregative for Schedule Risk]
11. ProductRisk=VagueAgg(SoftRelRisk,DBSizeRisk, ProcessComplexityRisk)
    [Apply Vague Aggregative for Product Risk]
12. PlatformRisk=VagueAgg(TimeBoundRisk, MemoryConsRisk,DBSizeRisk, MachineVolRisk)
    [Apply Vague aggregative for Platform Risk]
13  PersonalRisk=VagueAgg(AnaCapRisk,AppExpRisk, LangExpRisk,ProCapaRisk, MacExpRisk,ModProgPracRisk)
    [Apply Vague Aggregative for Personal Risk]
14. ProcessRisk=VagueAgg(ToolUseRisk, TimeConsRisk,TurnAroundTimeRisk, ProgCapRisk,MachExpRisk)
    [Apply Vague Aggregative for Process Risk]
}
}
```

The Algorithm 1 has formulated the method to extract the project features and trained them under various mathematical rules to analyze the quality of a software project. At the first level, the feature driven analysis on each individual rule is applied. Later on, composite evaluation of the software project is done by combining these high level rules. The algorithmic process has provided the product level, process level, personal level, platform level quality of software systems. The implementation of the proposed work model is done on Cocomo NASA 2 projects dataset. The implementation results are discussed in the next section.

## RESUTS AND ANALYSIS

In this paper, a vague inspired software quality estimator is defined based on various integrated features. The proposed model is implemented in matlab environment on PROMISE Software engineering repository. It is the publicly available repository which is having the description of 93 NASA projects designed between 1971 and 1987. The description of this collected dataset is shown in table 1

Table I. **Table 1 :Dataset Properties**

| Properties | Values |
|---|---|
| Dataset Repository | PROMISE Software Engineering Repository |
| Source of Data | http://promise.site.uottawa.ca/SERepository |
| Number of Projects | 93 |
| Year of Development | 1971 to 1987 |
| Number of Project Features | 24 |

Here figure 1 has shown the basic characterization of the dataset considered in this work for the analysis. Each of the projects is defined here with 24 different features. These features include the descriptive features such as name of project, project category, development environment, development year etc as well as the quality specific features including the storage support, platform support, tool support etc. The evaluation results based on various quality aspects are presented in this section. Figure 3 is showing the quality estimation based on the process level complexity of software projects.

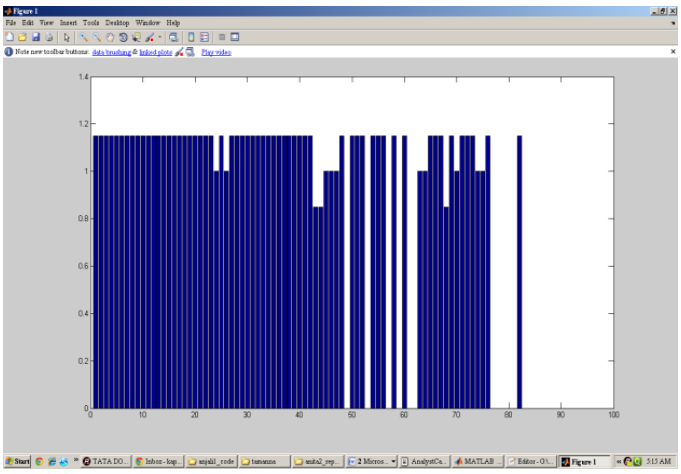

Figure 3. **Figure 3 : Process Complexity Analysis**

Figure 3 is shows the individual project evaluation based on process based evaluation. Here x axis is represents the software projects and y axis shows the complexity to represent the criticality in software process. The software process depends on multiple vectors including the machine

experience of programmer, tool usage experience of programmer, turn a round time o n software p roject and t he time constraint associated to the particular project. The white lines are s howing t he projects w ith l esser process l evel quality.

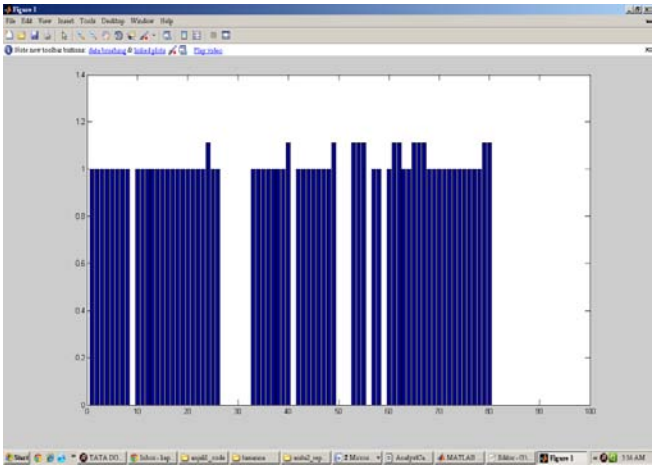

Figure 4. **: Personal Risk Analysis**

Figure 4 is shows the individual project evaluation based on personal evaluation. Here x axis represents the software projects and y axis shows the personal associated complexity to represent the criticality in software projects. The personal capability r epresents th e individual c apability in t erms o f language experience, machine experience, modified program practice e xperience et c. H igher t he personal quality o f software individuals, more reliable the software pr oject can be developed. The white lines are showing the projects with lesser personal level quality.

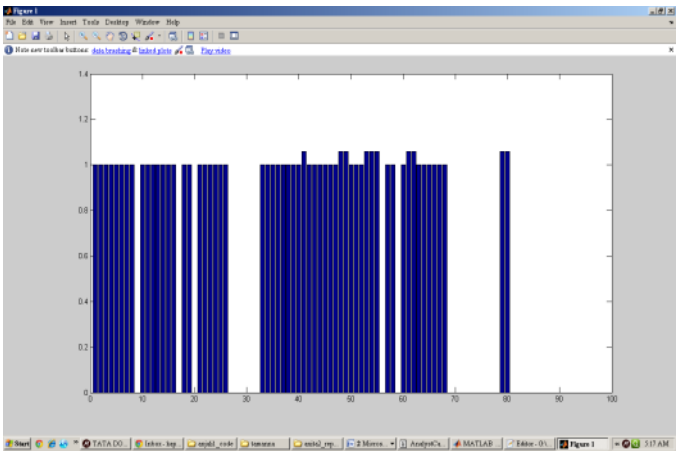

Figure 5. **: Platform, Risk Analysis**

Figure 5 is shows the individual project evaluation based on platform capability evaluation on which the software will be developed. H ere x a xis represents t he s oftware projects and y a xis s hows th e p latform a ssociated complexity concerns. T he platform quality aspect depends on the dead line s pecification, u ser s ystem memory, D B s ize a nd th e machine vo latility. I f t he developing environment a nd the user e nvironment a re not s ame, t he hi gher r isk i n s oftware delivery an d m ore chances o f s oftware m aintenance. S uch system can be c onsidered w ith lesser de liverable q uality features.

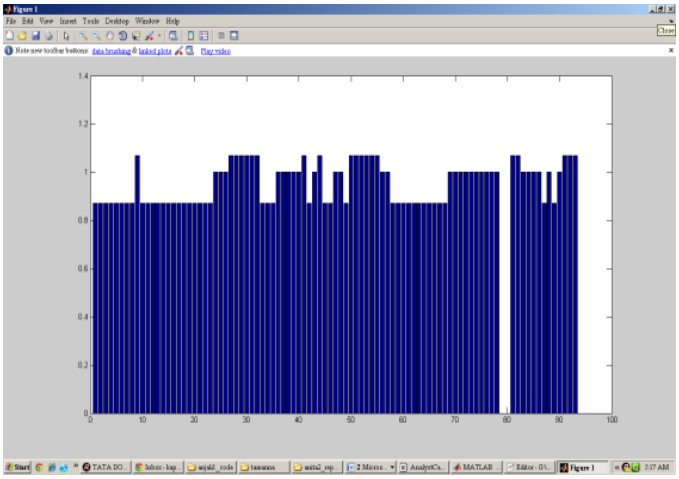

Figure 6. **: Product Risk Analysis**

Figure 6 is shows the individual project evaluation based on product c apability e valuation on w hich th e s oftware w ill be developed. H ere x a xis represents t he s oftware p rojects and y a xis s hows t he pr oduct a ssociated complexity concerns. T he pr oduct qu ality depends on the software release specific constraints, software size evaluation and the process level ex pectations. S oftware product i s t he a ctual deliverable s o that the complexity a t p roduct l evel m ust satisfy the high quality measures.

## CONCLUSION

In this paper, a vague inspired method is defined for software risk evaluation. The proposed work model first divided the available software features in relative categories. The method also evaluated th e in dividual f eature u nder m athematical rules as well as aggregative c ategory inclusive features. The work model is applied on Nasa 2 projects dataset. The results are obtained in terms of different risk categorization for all available projects.

## REFERENCES


[1] Reham E jaz," A Q uality Assurance M odel f or A nalysis Phase", NSEC'10, 04-OCT-2010, Rawalpindi, Pakistan ACM 978-1-4503-0026-1/10/10

[2] Philip Koopman," Risk Areas In Embedded Software Industry Projects", WESE'10, October 24, 2010, Scottsdale, AZ, USA. ACM 1-58113-000-0/00/0010

[3] Victor R. Basili," Obtaining Valid Safety Data for Software Safety M easurement an d Process I mprovement", ESEM'10, September 16–17, 2010, Bolzano-Bozen, Italy. A CM 978-1-4503-0039-01/10/09

[4] Emad Shihab," An Industrial S tudy on the R isk of S oftware Changes", SIGSOFT'12/FSE-20, N ovember 1 1–16, 20 12, Cary, North Carolina, USA. ACM 978-1-4503-1614-9/12/11

[5] G. Antoniol," Requiem for software evolution research: a few steps t oward t he cr eative ag e", IWPSE'07, S eptember 3 -4, 2007, Dubrovnik, Croatia. Copyright 2007 ACM ISBN 978-1-59593-722-3/07/09

[6] Piotr To maszewski," Comparing the Fault-Proneness of New and Modified Code — An Industrial Case Study", ISESE'06, September 2 1–22, 2006, R io de J aneiro, B razil. ACM 1 -59593-218-6/06/0009

[7] Shareeful I slam," Software D evelopment R isk M anagement Model – A G oal D riven A pproach", ESEC/FSE D octoral Symposium'09, Aug. 25, 2009, Amsterdam, The Netherlands. ACM 978-1-60558-731-8/09/08.

[8] Darko Du risic," Measuring t he S ize o f Changes i n Automotive S oftware S ystems a nd t heir I mpact on Product Quality".

[9] Tokuro M atsuo," Diversification o f R isk b ased on D ivided Tasks in Large-scale Software System Manufacture", June 12, 2007. San D iego, C alifornia, US A CM 978 -1-59593-856-5/07/06

[10] Shareeful I slam," Human F actors i n S oftware S ecurity R isk Management", LMSA'08, M ay 11, 2008, L eipzig, G ermany. ACM 978-1-60558-027-9/08/05

[11] Qi L i," A V alue-Based R eview P rocess f or Prioritizing Artifacts", ICSSP'11, M ay 2 1–22, 2 011, W aikiki, H onolulu, HI, USA ACM 978-1-4503-0730-7/11/05

[12] Kohl W itmer," A M ethod f or R isk M itigation D uring t he Requirements P hase f or M ultimedia S oftware S ystems", SIGDOC'06, October 18 –20, 2006, M yrtle B each, S outh Carolina, USA. ACM 1-59593-523-1/06/0010

[13] Idongesit M kpong-Ruffin," Quantitative S oftware S ecurity Risk Asse ssment M odel", QoP'07, O ctober 29 , 2007, Alexandria, V irginia, U SA. A CM 978 -1-59593-885-5/07/0010

[14] John D hlamini," I ntelligent R isk M anagement T ools f or Software D evelopment", S ACLA ' 09, 2 9 J une - 1 J uly, Mpekweni B each R esort, S outh Africa A CM 978 -1-60558-683-0/09/07

[15] Lucas L ayman," Mining S oftware Ef fort D ata: Preliminary Analysis of Visual Studio Team System Data", MSR'08, May 10–11, 2 008, L eipzig, G ermany. ACM 978 -1-60558-024-1/08/05