An Impact-based Analysis of Software Reengineering Risk in Quality Perspective of legacy System

Er. Anand Rajavat
[1]Department of Computer Science & Engineering
SVITS, Indore, M. P., India

Dr. (Mrs.) Vrinda Tokekar
Information Technology
IET (DAVV), Indore, M. P., India

ABSTRACT
Reengineering of operational legacy system is a novel technique for software rejuvenation. Reengineering is used specifically to satisfy and even delight modern customers and market with the value of our software products and services to gain their loyalty and repeat business. However, it incurs some overhead in terms of risk. The basic necessity for the successful implementation of reengineering strategy is to measure the overall impact of different reengineering risk components that arises from system, managerial and technical domain of legacy system. Quantifiable risk measures are necessary for the measurement of reengineering risk to take decision about when the evolution of legacy system through reengineering is successful. We present a quantifiable measurement model to measure comprehensive impact of different reengineering risk arises from quality perspective of legacy system. The model consists of five reengineering risk component, including Maintainability risk, Project complexity risk, Software architecture risk, Training Risk and Security risk component .Proposed measurement model offers better performance in terms of risk measurement to support the decision-making process.

Keywords
Reengineering, Risk Engineering, Measurement.

1. INTRODUCTION
Legacy systems [1] and the data they process are vital assets for the organization that use them. However, over the years many changes have been incorporated in the system that caused progressive degradation in the system quality.

This degeneration, together with the continual and often evolutions of the market and user requirements, leads to a high number of reasons for legacy system evolution. Over the past few years, legacy system re-engineering has emerged as an important system evolution strategy. A goal of software re-engineering is to take an existing software system and generate from it a new system that has the same quality as software created by modern software engineering practices [2].Reengineering is necessary, indeed indispensable, to overcome many of the most serious flaws of legacy system. In addition to improving the quality of the system, the reengineering process should enable new functions to be introduced and new technologies to be adopted, to satisfy current needs of business environment [3].Unfortunately, most of the reengineering projects are only concerned on satisfaction of current needs of business environment and usually ignore the reengineering risk components and factors that will affect quality attributes in the evolution process of the legacy systems. A feasible re-engineering process required to measure overall impact of different reengineering risk engenders from system, managerial and technical domains of legacy system [4] [5].

Proposed measurement model analyze current state of legacy system and desired state of target system to quantify different reengineering risk components arises from quality perspective of legacy system. A pentagram model is used to compute comprehensive impact of all the risk components. The quality perspective risk measurement model consists of five reengineering risk components, including Maintainability risk, Project complexity risk, Software architecture risk, training Risk and Security Risk component. Different Measurement metrics are used to examine and analyze different reengineering risk components. .

2. RELATED WORK
As with any engineering discipline, software reengineering requires a measurement mechanism for feedback and evaluation of reengineering risk. Reengineering risk measurement is a mechanism for creating estimation in answering a variety of questions associated with the enactment of any legacy system. Measurement allows us to determine the strengths and weaknesses of the legacy system that means what is the success rate of evolution strategy used to modernize legacy system.

Victor R. Basili in [6] develops a Goal Question Metric mechanism for defining and interpreting operational and measurable software. It can be within the context of a more general approach to software quality improvement. Whereas Linda Westfall in [7] develops a Kiviat chart to summarize different set of metrics. Kiviat chart can be used to compare several different items like projects, products or Processes across several parameters against the ideal. Yennun huang in [8] present a model for analyzing software renovation in running application and express downtime and costs due to downtime during renovation using some parameters in that model. On the other side Kishor S. Trivedi in [9] presents a stochastic model to measure the effectiveness of proactive fault management in software systems and determine optimal times to perform rejuvenation, for different scenarios. Model develops different methodologies to detect software aging and estimate its effect on various system resources.

The existing risk measurement models used for the evolution of legacy system really measure risk impact by considering current state of legacy system and desired state of target system. They are all based on the lower-level metrics needed in order to obtain satisfactory measures of higher-impact risk components. Henceforth it is required to measure comprehensive impact of all reengineering risk components in the evolution process of legacy system. The precondition to measure the comprehensive impact of reengineering risk is, to

measure impact of each risk component using different measurement metrics. Finally we need to integrate these risk components into a unified approach to get total risk impact..

3. QUALITY PERSPECTIVE RISK MEASUREMENT MODEL

The purpose of measurement model is to design quantifiable metrics for the evaluation of legacy system by measuring different reengineering risk components materialize in the evolution process of legacy system. Finally model is able to design quantifiable metrics to measure comprehensive impact of all reengineering risk arises from system, managerial, and technical domains of legacy system.

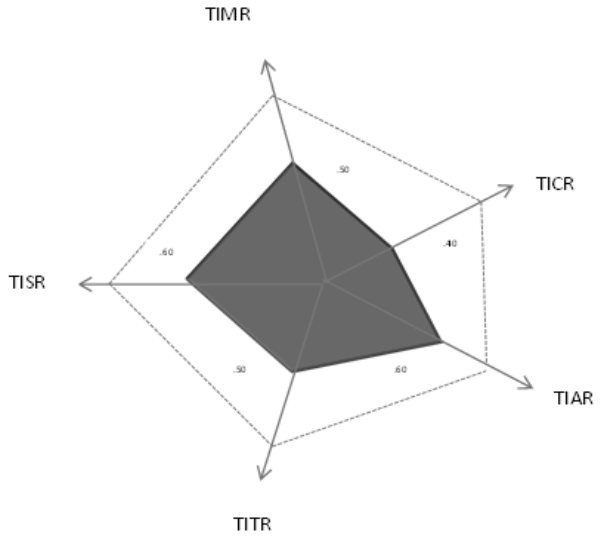

Figure 1 Quality perspective measurement model

As shown in Figure 1 quality perspective measurement model is presented through using a pentagram diagram based on the measurement of its five risk components. The total impact of each risk component is measured based on the results of their metrics during measurement process. Let us assume: the measurement results of each risk component is a value from 0 to 1. The value "1" indicates the maximum value for each risk component, and "0" indicates the minimum value. The area of the pentagram is used as the measurement of overall impact of five risk components. Clearly, the smallest value of this pentagram area is 0, and the maximum value is approximately 2.4. As the pentagram consists of five triangles, the area of each triangle can be computed 0.5 * L1 * L2 * Sinα where L1, L2 represent the sides of the triangle and α represents the 72-degree angle between the two sides. The term TIMR, TICR, TIAR, TITR, and TISR in Figure 1 are used to represent the five risk components of Model respectively. Since each risk component is measured using different quantifiable measurement metrics the Total Risk Impact (TRI) of all risk components can be computed as below:

$$\text{Total Risk Impact (TRI)} = 1/2 \sin 72\,(ab + bc + cd + de + ea)$$

$$= \tfrac{1}{2} X\, 0.9511X(ab + bc + cd + de + ea)$$

$$= 0.48\, X\,(ab + bc + cd + de + ea)$$

Where a represents TIMR, b represents TICR, c represents TIAR, d represents TITR, e represents TISR

The main advantages of pentagram area calculation are

- In the pentagram model, every estimate is equal and independent.
- It adopts subjectively and objectively integrative estimation.
- It is easy to show the difference of the sum. If one dimension has a lesser change, and the other four have no change, the area calculation method with the influence on total scores will be more than weighted average computing [10].

Quality perspective risk measurement model is used to measure overall impact of five different reengineering risks from technical domain of legacy system. The five most important risk components of quality perspective risk measurement model are presented in Table I. It shows Key risk component and the most important measures of those components. The purpose of quality perspective risk measurement is to design a pentagram model used to measure overall impact of different reengineering risk component in quality perspective of legacy system.

Table 1 Most important measure

Key Risk component	*Most Important Measures*	*Symbol*
Maintainability Risk	*Backlog management index (BMI)* *Fix response time (FRT)* *Percent delinquent fixes (PDF)* *Fix quality (FQ)*	*a*
Project complexity risk	*Cyclomatic complexity* *Design Complexity* *Global data Complexity* *Data complexity*	*b*
Software architecture Risk	*Architecture adaptability index (AAI)* *Software adaptability index (SAI)*	*c*
Training Risk Cluster	*level 1 evolution Reaction* *level 2 evolution Learning* *level 3 Behavior* *level 4 Results*	*d*
Security Risk	*Threat* *Vulnerability* *Cost* *no. of modules in legacy system*	*e*

4. MEASUREMENT METRICS

In this section, we describe the measurement metrics used to measure impact of each risk component represented by each side of the pentagram model

4.1 Maintainability Risk

Maintainability is the ease with which a program can be correct if an error occurs. When the development of a software product is completed and it is released to the market, it enters the maintenance phase of its life cycle. During this phase the defect arrivals by time interval and customer problem calls (which may or may not be defects) by time interval are the major issues. The main task during the maintenance phase is to fix the defects as soon as possible and with excellent fix quality. Such actions, although still not able to improve the defect rate of the product, can improve customer satisfaction to a large extent. Maintainability risk component is the probability that the reengineered system facilitate updates to satisfy new requirements in future. Maintainability risk measurement model measure that the software product that is maintainable should be well-documented, should not be complex, and should have spare capacity for memory, storage and processor utilization and other resources.

The following measurement metrics is used to measure the total impact of maintainability risk component.

TIMR = BMI+FRT+PDF+FQ

Where

BMI represents backlog management index

FRT represents fix response time

PDF represents Percent Delinquent Fixes

FQ represent fix quality

- Backlog management index (BMI)

Fix backlog is a workload statement for software maintenance. It is related to both the rate of defect arrivals and the rate at which fixes for reported problems become available. It is a simple count of reported problems that remain at the end of each month or each week. Metric to measure the backlog problems is the backlog management index (BMI).

$$BMI = \frac{\text{Number of problems closed during the month}}{\text{Number of problem arrivals during the month}*100\%}$$

As a ratio of number of closed, or solved, problems to number of problem arrivals during the month, if BMI is larger than 100, it means the backlog is reduced. If BMI is less than 100, then the backlog increased.

- Fix Response Time (FRT)

For many software development organizations, guidelines are established on the time limit within which the fixes should be available for the reported defects. Usually the criteria are set in accordance with the severity of the problems. For the critical situations in which the customers' businesses are at risk due to defects in the software product, software developers or the software change teams work around the clock to fix the problems. For less severe defects for which circumventions are available, the required fix response time is more relaxed. The fix response time metric is usually calculated as follows for all problems as well as by severity level:

Mean time of all problems from open to closed

- Percent Delinquent Fixes (PDF)

The mean (or median) response time metric is a central tendency measure. A more sensitive metric is the percentage of delinquent fixes. For each fix, if the turnaround time greatly exceeds the required response time, then it is classified as delinquent:

$$PDF = \frac{\text{Number of fixes that exceeded the response time criteria by severity level} * 100\ \%}{\text{Number of fixes delivered in a specified time}}$$

- Fix Quality (FQ)

Fix quality or the number of defective fixes is another important quality metric for the maintenance. From the customer's perspective, it is bad enough to encounter functional defects when running a business on the software. It is even worse if the fixes turn out to be defective. A fix is defective if it did not fix the reported problem, or if it fixed the original problem but injected a new defect. For mission-critical software, defective fixes are detrimental to customer satisfaction. The metric of percent defective fixes is simply the percentage of all fixes in a time interval (e.g., 1 month) that are defective [11].

4.2 Project complexity risk

Software complexity is defined as an important determinant of software maintenance risk. Increased software complexity means that maintenance and enhancement projects will take longer, will cost more, and will result in more risk. Some metrics are proposed to measure the complexity of software. Cyclomatic complexity may be considered a broad measure of soundness and confidence for the measurement of project complexity risk. It measures the number of linearly-independent paths through a program module. This measure provides a single ordinal number that can be compared to the complexity of other programs. Project complexity risk component is the risk of loss associated with the complex legacy system functions that are hard to evolve through reengineering. Project complexity risk measurement Model measures complexity of different functions of legacy system

with the help of different software complexity measurement tools.

The total impact of Project complexity risk (TICR) can be measured using following metrics

TICR= CC+MDC+GDC+SDC

Where

CC represents Cyclomatic complexity
MDC represents Module design complexity
GDC represents Global data complexity
SDC represents Specified data complexity

- Cyclomatic complexity

Cyclomatic complexity is a measure of the logical complexity of a module and the minimum effort necessary to qualify a module. Cyclomatic is the number of linearly independent paths and, consequently, the minimum number of paths that one should (theoretically) test.

- Module Design Complexity

Module design complexity of a module is a measure of the decision structure which controls the invocation of the module's immediate subordinate modules. It is a quantification of the testing effort of a module as it calls its subordinates. The module design complexity is calculated as the cyclomatic complexity of the reduced graph. Reduction is completed by removing decisions and nodes that do not impact the calling control of the module over its subordinates.
Important factors of Module design Complexity

- Modules do not exist in isolation
- Modules call child modules
- Modules depend on services provided by other modules

- Module Global Data Complexity

Global data complexity quantifies the complexity of a module's structure as it relates to global and parameter data. Global data is data that can be accessed by multiple modules. This metric can show how dependent a module is on external data and is a measure of the testing effort with respect to global data. Global data complexity also measures the contribution of each module to the system's data coupling, which can pinpoint potential maintenance problems.

- Module Specified Data Complexity

Specified data complexity quantifies the complexity of a module's structure as it relates to user-specified data. It is a measure of the testing effort with respect to specific data. Data dictionary is used to select a single data element, all elements with a specific data type, or a variety of other selection criteria. The specified data complexity then quantifies the interaction of that data set with each module's control structure [12].

Structural Analysis

The higher the complexity the more risk. The more risk the more risky to evolve legacy system using reengineering strategy. Correlation between Cyclomatic complexities with Reliability & maintainability Risk and Bad fix probability is shown in the table 2

Table 2 Structural analysis

Cyclomatic Complexity	Reliability Risk
1-10	Simple procedure, little risk
11- 20	More complex, moderate risk
21 – 50-	Complex, high risk
>50	Untestable, very high risk
Cyclomatic Complexity	Bad Fix Probability
1 – 10	5%
20 –30	20%
> 50	40%
Approaching 100	60%
Cyclomatic Complexity	Maintainability Risk
1 – 4	Structured, little risk
> 4	Unstructured, high risk

4.3 Software architecture Risk

Software architecture is typically documented using multiple views. A "view" is a representation of a set of system elements and the relationships associated with them. Together, these definitions are saying that the software architecture serves multiple purposes and hence cannot be captured in a single model (i.e., a view).Any software architecture has two elements – components and connectors. Software could be adaptable with respect to either of these two elements on any of the architectures for that software. With respect to software adaptability, as defined above, adaptability of either element carries equal significance or weight.

Software architecture Risk Component is the risk of loss associated with inconsistency between existing and desired architecture of legacy and target system. Software architecture risk measurement model analyzes legacy systems structure, comprising software elements, the externally visible properties of those elements, and the relationships between them in accordance with architecture of target system. Identification and resolution of architectural risk is one of the key factors in successfully reengineering effort. Architectural consistency risk often leads to project inefficiencies, poor communication, and inaccurate decision making. Identifying and controlling architectural risks can have a significant impact on the overall success of a reengineering effort [13].

Total impact of software architecture risk (TIAR) can be computed using following metrics

TIAR= AAI+SAI

Where AAI represents Architecture adaptability index (AAI)

SAI represents Software adaptability index (SAI)

$$\text{Architecture adaptability index (AAI)} = \frac{\text{EAI for all elements of architecture}}{\text{Total number of elements}}$$

$$\text{Software adaptability index (SAI)} = \frac{\text{AAI for all architecture of the software}}{\text{Total number of architectures for that software}}$$

Legacy Systems

In many cases, especially legacy code systems, only the source code is available. In such cases using tools the code architecture may be obtained. For software systems that have only the code architecture,

SAI = AAI (code architecture).

An adaptable element of architecture has a unit element adaptability index (EAI). A non-adaptable element of architecture has zero EAI.

4.4 Training Risk

Training risk component is the risk of loss associated with the lack of training for the existing work force on advanced tools and technology which will be used to achieve target system goals. Training risk measurement model measure the requirements of customized and specialized training programs and special consulting services for present user of the legacy system so that they are comfortable with operations of target system. The Training risk identifies the key elements and steps necessary for training the various staff to use of the relevant functionality of the target system.

Training effectiveness means how well the training inputs are serving the intended purpose. This aspect is often neglected by organizations, saying that measurement is difficult. There are three kinds of training outputs that organizations need to measure. They are:

- Relating to evolution planning, relevance, comprehension and whatever defined goals in the evolution process.
- Utilization ratio for enhanced functionality provided by the target system
- Redeveloping legacy system in to target system by considering skills competencies, decision making and problem-solving abilities of the organization
- The changes in the mind set such as work related attitudes, values, interpersonal competencies and personal attributes towards target system.

The total impact of training risk component can be calculated using following measure

TITR=R1+L+B+R2

Where

R1 represents level 1 evolution Reaction
L represents level 2 evolution Learning
B represents level 3 Behavior
R2 represents level 4 Results

- Reactions

This, the first "level" consists of information about the trainees' perceptions of the training: Do they see it as use-full? Was their time well spent? Did the instructor know what he or she was doing? All kinds of questions are asked and answered as part of evaluating training at this level. A training course that consistently and repeatedly fails the level1 is in deep trouble. If trainees uniformly and consistently dislike or claim that a particular training session (or instructor) is of little or no value to them, it is doomed. The level1 must be passed.

- Learning

This level deals with the acquisition of skill and knowledge during the training, usually as evidenced by en-route and end-of-course assessments. Can the trainees do what they're being trained to do? When used in conjunction with pre-tests, this kind of assessment can do a reliable job of determining if the training course or session is achieving its learning objectives. It is, then, a reasonably good measure of the efficiency of the training.

- Behavior

Behavior change on the job is indeed another important measure; however, it is as much or more a measure of two other factors than it is of training itself. One factor is the extent to which the job environment supports applying what was learned and the second factor is the applicability and utility of what was learned.

- Results

The level 4 focuses on results in the workplace. These might be operational such as reduced errors or increased productivity and they might be financial such as reduced costs or increased sales. But for training to lay claim to the credit for any results in the workplace there is a bridge that must be constructed spanning the gap between behavior changes and business results. Finding the links between changes in human behavior and changes in business results can be a taxing and difficult task. It requires being able to identify the linkages connecting the two and that requires being able to work your way through the performance architecture of the organization [14].

4.5 Security Risk

Security can be defines as ability to protect data against unauthorized access and to withstand malicious or inadvertent interference with its operations. Security ensures that important data and information of present system is not accessed by unauthorized persons during system evolution process. Security risk component is the probability that the important data and information of existing system is lost or misused during system evolution process. Security risk measurement model measures the effectiveness of the process that will ensure the availability and confidentiality of important data and information of legacy system after the evolution through reengineering.

Security in a software-intensive system is achieved and preserved by a wide range of activities associated with the original system development process and with system operations in the use environment, including maintenance. Security is defined in risk terms (likelihood and severity of failures); all security related activity can be regarded as forms of risk reduction. Strengthening engineering practices reduces the risk of introducing faults during development. Engineering security architecture and security-specific components reduces the risk of failures due to common types of faults. Assurance activity reduces uncertainty about risks, especially for users and others not directly involved in development.

Two specific risk assessment activities are important in the security field: threat modeling and vulnerability assessment.

- Threat modeling-

Factors involved in assessing the security risk posed by a particular agent have been modeled in [15]. These factors can be assessed on the basis of qualitative scales, enabling risks to be prioritized.

- Vulnerability Assessment-

Assessing system and software designs and implementations for potential vulnerabilities complements the threat-driven approach. For software, vulnerability scanning tools are available today to assist with the detection of defects commonly associated with security events. The tracking of potentially exploitable defects and vulnerabilities enables the measurement of numbers of these over time, in different type and status categories. In practice false positives can be a severe problem, for legacy code.

The total impact of Security risk component can be calculated using following metrics

$$(TISR) = \sum_{k=0}^{n} (T \times V \times C)$$

Where

T represents Threat
V represents Vulnerability
C represents Cost
K represents no. of modules in legacy system

Threat is the frequency of adverse events. Vulnerability is the likelihood that a particular attack will be successful, and cost is the total economic impact of a successful attack.

5. EXPERIMENT AND ANALYSIS

The purposes of this experiment is to check the correctness of the referred quantitative risk measurement model for quality perspective of legacy system and compare the result of two legacy software of the same scenarios i.e. library management system. The two different legacy library management systems are used to check the correctness of proposed measurement model. The total impact of five identified risk is calculated for each legacy library management system using different measurement metrics for respective risk components.

Applying pentagram model to five risk components we have the following results as shown in table 3.

TRI (LS1) = 0.48 X $(ab + bc + cd + de + ea)$

=0.48 X [(.35*.20) + (.20*.26) + (.26*.50) + (.50*.40) + (.40*.35)]

=.284

TRI (LS2) = 0.48 X $(ab + bc + cd + de + ea)$

=0.48 X [(.50*.40) + (.40*.60) + (.60*.50) + (.50*.60) + (.60*.50)]

=.643

Where LS1 represents legacy system 1

LS2 represents legacy system 2

TRI Total Risk Impact

Table 2 Results

System	a	b	c	d	e	TRI
LS1	.35	.20	.26	.50	.40	.284
LS2	.50	.40	.60	.50	.60	.643

Based on the TRI values of two legacy library management system the measurement results for both systems i.e. LS1 and Ls2 tests are shown in Figure 2. It is clear that the TRI of LS2 (Right) is higher than the TRI of LS1 (Left).

A mean opinion score is used to quantify and predict the judgment based on total impact reengineering risk from quality perspective of legacy system.

Table 3 Mean opinion score

Level of Satisfaction	Risk Impact	Range of values
Reengineering successful	Low TRI value	0-1
Need Risk Resolution strategy	Average TRI value	1-1.5
Massive Risk engineering /Reengineering failure	High TRI value	1.5-2.4

We give comparative TRI values for LS1 and LS2 with the mean opinion score. The mean opinion scores and the corresponding measurement model based impact values are shown on the table 4. It shows that reengineering is successful if the TRI value is less than or equals to 1 and the values higher than this level required massive risk engineering or tends to reengineering failure.

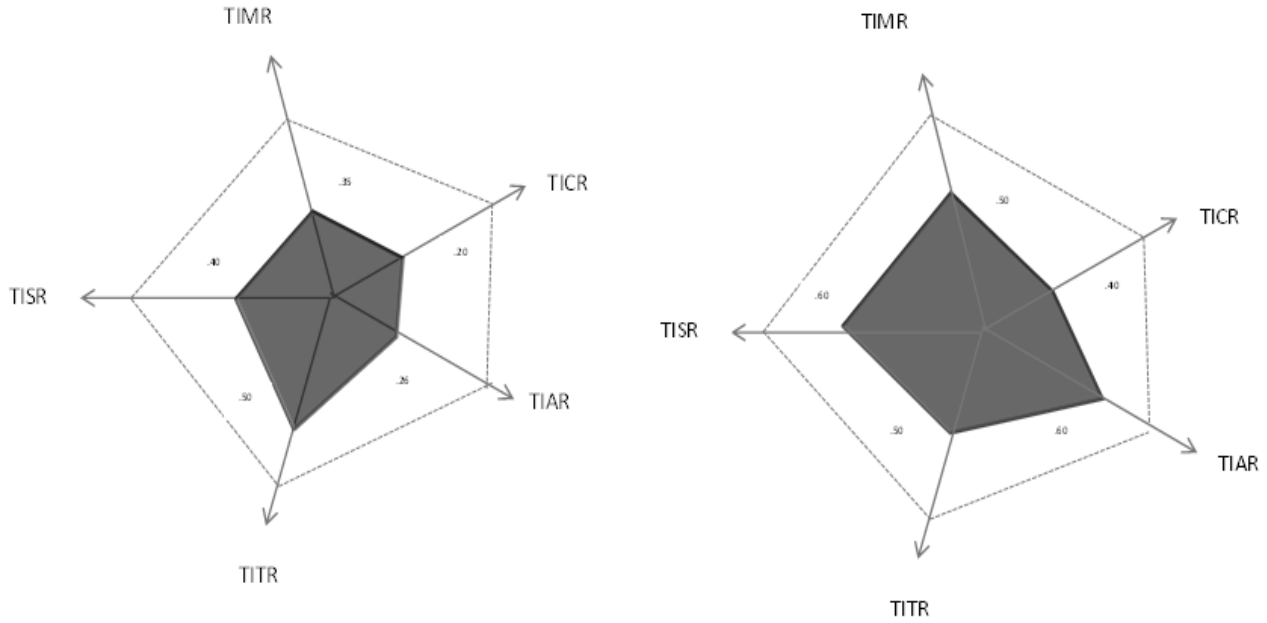

Figure 2 Comparative analysis

6. CONCLUSION

Software reengineering is a revolutionizing technique used to modify structure and values of the systems data. Reengineering involves analysing the current state of the legacy system and transforming the system state according to the requirements of target system. The purpose of introducing risk measurement in reengineering process is to facilitate the developers and client of legacy system to be aware and measure total impact of all the risk that could be identified in system, managerial and technical domains of legacy system. In this paper, we propose a measurement-based model to estimate the comprehensive impact of reengineering risk arises from quality perspective of legacy system. We also use referred model to compare total risk Impact (TRI) for two legacy systems of the same domain. These results are the basic to lay the foundation for the inception of a decision system to facilitate software reengineering as a system evolution strategy. Finally a mean opinion score board is developed based on TRI value to support decision making system for a successful reengineering solution.

7. REFERENCES

[1] Brodie, M. L., Stonebraker, M., "Migrating Legacy Systems: Gateways, Interfaces, & the Incremental Approach," Morgan Kaufmann Publishers, Inc.; 1995.

[2] Byrne, E.J. Gustafson, D.A.," A software re-engineering process model", in Proceeding of, Sixteenth Annual International Conference on Computer Software and Applications, Digital Object Identifier: 10.1109/CMPSAC.1992.217608 , ISBN: 0-8186-3000-0 ,1992 , PP 25-30.

[3] Bianchi, A., Caivano, D., Marengo, V., Visaggio, G.," Iterative reengineering of legacy functions", Proceeding of IEEE International Conference on Software Maintenance, Digital Object Identifier: 10.1109/ICSM.2001.972780, ISBN: 0-7695-1189-9 2001, PP 632-641.

[4] Anand Rajavat, Dr. (Mrs.) Vrinda Tokekar, "SysRisk –A Decisional Framework to Measure System Dimensions of Legacy Application for Rejuvenation through Reengineering", Published in International Journal of Computer Applications (IJCA), 16(2):16–19, February 2011, ISBN: 978-93-80747-56-8, Doi 10.5120/1985-2674.

[5] Anand Rajavat, Dr. (Mrs.) Vrinda Tokekar, "ReeRisk –A Decisional Risk Engineering Framework for Legacy System Rejuvenation through Reengineering", Published in Proceedings of Second International Conference on Recent Trends in Information, Telecommunication and Computing – ITC 2011 by Springer LNCS-CCIS, March 10-11, 2011 in Bengaluru, India, CNC 2011, CCIS 142, pp. 152 – 158, 2011, © Springer-Verlag Berlin Heidelberg 2011.

[6] Victor R. Basili1, Gianluigi Caldiera1 H., Dieter Rombach," The Goal Question Metric Approach", technical report, department of computer science, institute for advanced computer studies, university of Maryland.

[7] Linda Westfall," Kiviat Charts", technical report, the west fall team, partnering for software excellence.

[8] Huang, Y.; Kintala, C.; Kolettis, N.; Fulton, N.D.," Software rejuvenation: Analysis, Modeling, and applications", Twenty-Fifth International Symposium on Fault-Tolerant Computing, 1995. FTCS-25. Digest of Papers, Digital Object Identifier: 10.1109/FTCS.1995.466961, 1995, Pp381-390.

[9] Trivedi, K.S.; Vaidyanathan, K.; Goseva-Popstojanova, K," Modeling and analysis of software aging and rejuvenation", Proceedings. 33rd Annual Simulation

Symposium, 2000, Digital Object Identifier: 10.1109/SIMSYM.2000.844925, 2000, pp270-279.

[10] Yan Gong; Fangchun Yang; Lin Huang; Sen Su," Model-Based Approach to Measuring quality Of Experience", First International Conference on Emerging Network Intelligence, Digital Object Identifier: 10.1109/EMERGING.2009.17,2009,PP:29-32.

[11] Stephen H. Kan," Metrics and Models in Software Quality", Addison-Wesley Professional, 2 nd Edition, ISBN-10: 0-201-72915-6, 2003.

[12] Thomas J. McCabe, "Design complexity measurement and testing", Communications of the ACM, doi 10.1145/76380.76382, Volume 32 Issue 12, Dec. 1989.

[13] Nary Subramanian, Lawrence Chung," Metrics for Software Adaptability", Technical report, Applied Technology Division, Anritsu Company, 1999.

[14] Robert O. Brinkerhoff, Dennis Dressler," Using evaluation to build organizational performance and learning capability: A strategy and a method", Article, Performance Improvement, DOI: 10.1002/pfi.4140410605, Volume 41, Issue 6, pages 14–21, July 2002.

[15] John Murdoch," Security Measurement", White Paper, V3.0 13 January 2006.