ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 7, July 2018

Enhancing the Performance of Software Success Failure Analysis using Decision Tree Algorithm

G. Paul Suthan, S. Amalraj

Vice Principal and HOD, Department of Computer Science, Bishop Appasamy College of Arts and Science, Coimbatore, Tamilnadu, India

M.Phil Scholar, Department of Computer Science, Bishop Appasamy College of Arts and Science Coimbatore, Tamilnadu, India

ABSTRACT: The enhanced demand for better technology and everlasting global expansion continue to afford developers with lot of project opportunities for success or failure. While no industry is protected from project failure, the Information Technology (IT) industry is shown to be more prone to risk and failure than those of other production industries. In this digital era, facing software problems is becoming a part of our daily life. Every hour and every minute, we are facing several software problems, including poor usability, application failures, errors in user documentation; security bugs availability and performance problems which are associated with IT services. Since IT systems are used everywhere such problems are very common.

Risky, Inherently complex, risky and need of careful planning are the three main characteristics of software reliability. If the project is properly planned, then it ensures that the project doesn't fail, whereas, customers will get a clear definition of the project, know the status of the project and also they will be ready to access the project deliverables at any point of time. Most recent surveys says that poor process of requirement analysis and testing inadequate planning and specifications, ill defined requirements, complexity and lack of metrics and measures to compute project's sheer size, all together lead to, numerous change requests, delays, significant added costs and increase in the possibility of errors.

KEYWORDS: SSAT, Software testing, Software Analysis, Software Requirements Analysis

I. INTRODUCTION

The enhanced demand for better technology and everlasting global expansion continue to afford developers with lot of project opportunities for success or failure. While no industry is protected from project failure, the Information Technology (IT) industry is shown to be more prone to risk and failure than those of other production industries. In this digital era, facing software problems is becoming a part of our daily life. Every hour and every minute, we are facing several software problems, including poor usability, application failures, errors in user documentation, security bugs, availability and performance problems which are associated with IT services. Since IT systems are used everywhere such problems are very common.

The software problems and defects cost annually about 59.5 billion the US economy as stated by The National Institute of Standards and Technology. The rework especially problem resolution and bug fixes in software projects show the way to higher software development, higher prices for IT services and products and maintenance costs. In addition to that the escalating number of problems and defects, the quality of the process of managing defects and problems needs to be improved. When we concern about the problems relating to quality, Problems concerning quality in a problem management process can lead, for example, following symptoms: end users need not know who is the right person to contact and wrongly report problems to the service desk; also they claim that the IT provider does not respond to the problems reported by customers; The number of open problems at the service desk is rapidly increases

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 7, July 2018

in relation to closed problems; and so the waiting time of customers have to wait for a long time for solutions to their problems.

It's an interesting question to analyze why the above problems related to quality is still exist in software Engineer. Software quality assurance methods were proposed by previous studies that including inspections [1, 2], defect management methods [3,4], risk management techniques [5], various testing approaches (module testing, integration testing, system testing, and acceptance testing) [6], pair programming [7] and reviews [8]. Those studies states that most of the methods were developed to assure software quality in product-oriented software development than business related to service oriented.

Need of Reliability in Software Development

The software development process which is aimed at prediction, analyzing, preventing and also mitigating failures over time to time. Reliability is quality which can be measured over time. Reliable, error-free software continues to gratify customers for a long time.

Reliability, in the field of software industry, is the probability that an application program that operates successfully for a specific period of time and under specified conditions. The assurance or improvement of reliability is an integrative process which occurs during development, production, field phases. This must be implemented as per the internal standards for each product designed. To save more money during product warranty time, the money could be invested for developing reliable software at the time of design and production. Product reliability analysis forms the basis for Warranty policy as today's customers are more concern about the product reliability than few years ago Primarily, Software industries will get more benefit from good reliability because its product has added customer appeal, through faster service, higher quality and lower support cost. Secondly, the Customer who got the reliable software is reducing his running cost, lower down-time and higher availability.

II. LITERATURE REVIEW

Risky, Inherently complex, risky and need of careful planning are the three main characteristics of software reliability. If the project is properly planned, then it ensures that the project doesn't fail, whereas, customers will get a clear definition of the project, know the status of the project and also they will be ready to access the project deliverables at any point of time. Most recent surveys says that poor process of requirement analysis and testing inadequate planning and specifications, , ill defined requirements, complexity and lack of metrics and measures to compute project's sheer size, all together lead to, numerous change requests, delays, significant added costs and increase in the possibility of errors.

The three important factors which avoiding software failures are perfect requirement analysis method, proper software complexity management and exact testing techniques. Based on these concepts, this thesis is organized.

Software Requirements Analysis

The key for successful projects is to understand the problem domain and its requirement. Institute of Electrical and Electronics Engineers (IEEE) defines Requirements as a "condition or a capability that must be met or possessed by a system, to satisfy a contract, standard, specification or other formally imposed document". Somerville and Sawyer define requirements as a "specification of what should be implemented". Understanding requirements plays a vital role in problem definition. Requirement Engineering encompasses systematic and repetitive techniques in discovering, modelling, documenting and maintaining a set of requirements for a system as a whole, or specifically for software components of a system.

Findings on failures due to Poor Requirement Analysis

Most of the studies that had been done recently, quantifies cost and causes of software failures. As discussed in the introduction, imperfect requirement gathering, analysis and management are the main cause of failures. Statistics says nearly 60% – 80% of project failures are related to the above said problem. In another report, it states that 68% of IT projects fail especially due to poor requirements**.** It is noted that companies pay a premium of above 60% on time and budget, when they follow improper requirements practices in their projects. When the IT companies are using average analysts, they spend over 41% of the IT development budget for software, staff and external professional services. Sloppy development practices are also an important source of failure and it can cause errors at any stage of an IT project. Moreover, errors that are introduced during requirements phase and fixed later in the Software Development Life Cycle (SDLC) increase the cost exponentially.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 7, July 2018

III. SOFTWARE SUCCESS ANALYSIS TOOL (SSAT)

For software development analysis various tools and platforms have been used till nowadays. SSAT software tool is used as the platform in this project. SSAT is a web based analysis tool for analyzing software development process. This tool helps the organization whether the new project going to be developed will succeed or fail.

For creating this SSAT tool, we have used Microsoft .Net Framework version 3.5 as frontend and Microsoft SQL Server 2008 as backend. Data as feedback collected from various stakeholders like developers, project manager, organization, project leader etc., are taken for analysis which is done by this tool. It uses Decision Tree Algorithm for properly analysing the results. In this chapter, we will see the software and algorithms used to develop this tool.

Objectives of SSAT

1. To provide a consistent object-oriented programming environment whether object codes is stored and executed locally on Internet-distributed, or executed remotely.
2. To provide a code-execution environment to minimizes software deployment and guarantees safe execution of code.
3. Eliminates the performance problems. There are different types of application, such as Windows-based applications and Web-based applications. To make communication on distributed environment to ensure that code be accessed by the SSAT tool can integrate with any other code.

IV. SYSTEM IMPLEMENTATION

In the previous chapters in this research work we are working to offer a model to detect the software development success / failure using Decision Tree Algorithm. And to answer this we have used the evaluated Dataset, which is used in the SSAT Statistical Environment.

Input design is the process of converting user-originated inputs to a computer understandable format. Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. A large number of problems with a system can usually be tracked backs to fault input design and method. Every moment of input design should be analyzed and designed with utmost care.

The design of the input should be made the input as the over to the numerous networks in the reliable area that should be passed as the installation in the remote network. It has the following constraints in the input database.

- All the files from the disk should be acquired by data.
- It is suitable to more available data clearance and made available.
- The menu of design should be understandable and it is in the right format.

The system takes input from the users, processes it and produces an output. Input design is link that ties the information system into the world of its users. The system should be user-friendly to gain appropriate information to the user.

The project gives the low time consumption to make the sensitive application made simple. The amount of fund that the company can spend into the research and development of the system is limited. The design of input covers all the phases of input from the creation of initial data to actual entering of the data to the system for processing.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 7, July 2018

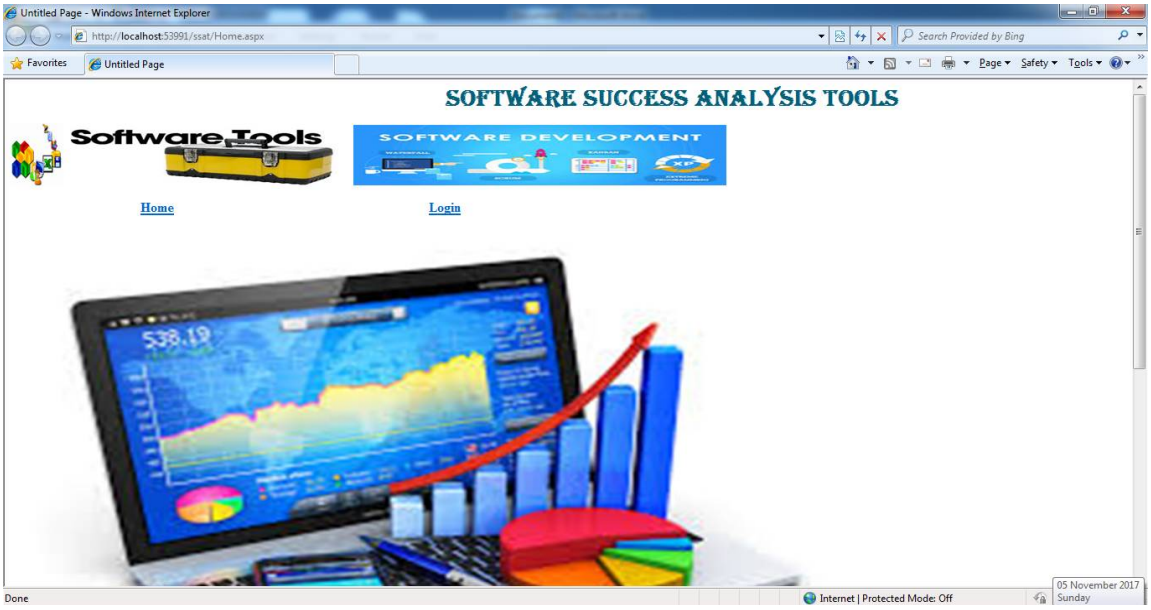

Fig.1. Software success Analysis Tools

The above figure displays the home page of Software Success Analysis Tools in a website. We are choosing Login link to proceed the success / failure analysis of a software development process before it can be started in an organization. Five users namely, Organization Admin, Project Manager, Project Leader, Software Developers and System Admin who generates the report can login using their Login Name and Password. The following questions were asked and the feedbacks were collected from different stakeholders.

V. CONCLUSION AND FUTURE SCOPE

In this thesis, we have discussed the three main factors of software failure. They are imperfect requirement analysis method, improper software complexity management and wrong testing techniques. Also, from the literature review, we have identified that most of the methods for identifying the software failure are fully collected after the failure occurred. So, getting feedback from the team leader before the commencement of such projects will reduce the failure costs. Based on this idea, the thesis is proposed.

For identifying and analyzing the input of the organization head, team manager, team leader and developer, Software Success Analysis Tool (SSAT) is created. This tool will collect the information from various stakeholders as said above, will be stored in database and the feedbacks are analyzed. Later on Decision tree Algorithm is applied to get the proper output and an average mark for each project given are calculated based on the weightage.

So, the final result of this tool suggest the management whether the project will succeed or not. If it gives negative results, then the cause of failure will also be analyzed and the tool will give suggestion to improve the areas to make that failure project, a successful one.

REFERENCES

1. M. Bush, , Los Alamitos, CA, "Improving software quality: the use of formal inspections at the jpl", In ICSE '90: Proceedings of the 12th international conference on Software engineering, pages 196–199, USA, 1990. IEEE Computer Society Press.
2. Tom Gilb and Dorothy Graham, "Software Inspection. Addison-Wesley", 1993.
3. Michael Fredericks, "Using defect tracking and analysis to improve software quality" US Air Force Research Laboratory Report SP0700-98-D-4000, 1998.
4. David N. Card, "Learning from our mistakes with defect causal analysis", IEEE Software, 15(1):56–63, January/February 1998.
5. JyrkiKontio, Software Engineering Risk Management: "A Method, Improvement Framework, and Empirical Evaluation", PhD thesis, Helsinki University of Technology, 2001.