Review of Software Reliability Analysis Models: A Case Study of Operating Systems

Muhammed B. Abdullahi, John Alhassan
Department of Computer Science
Federal University of Technology
Minna, Nigeria

Adepoju Solomon, Abubakar Adamu*
Department of Computer Science
Federal University of Technology
Minna, Nigeria
*Email: dotab2003[AT] yahoo.com

***Abstract*— Software need to be reliable in order to be trustworthy and dependable. Reliability analysis is one of the most important factors in software development since analyzing reliability of software during design and prior to release significantly save cost of failure testing activities. To this end, most of the method used for Software reliability analysis focused on product of large server in which the reliability is measured in terms of failure only, in this case, failure data is collected manually by service organization. Such method cannot be used to analyze reliability of operating system since it run on many operational profiles and manual data collection will be inadequate. Software reliability analysis requires thorough integration of set of reliability modeling, allocation, estimation, prediction and test task. In this research, we present three approaches that we can use to analyze Operating system; systematic quality assurance process, quantitative measurement and reliability feedback data. These are collectively used for analyzing reliability. We realized that when these three approaches are used collectively to analyze reliability of operating systems, it will lead to improvement in quality, dependability, reliability, usability, confidentiality, performance and durability.**

Keywords- Software reliability, analysis, operating system, defect, failure.

I. INTRODUCTION

To this end, most of the method used for Software reliability analysis focused on product of large server in which the reliability is measured in terms of failure only, in this case, failure data is collected manually by service organization. Such method cannot be used to analyze mass market s/w reliability such as windows and Linux operating since they run on many operational profiles and manual data collection will be inadequate. Software reliability analysis requires thorough integration of set of reliability modeling, allocation, estimation, prediction and test task.

Operating system must be reliable in order to meet customer satisfaction and establish user confidence. To analyze operating system, one must not define what will not be collected, must not collect what will not be analyze and must not analyze what will not be used. Operating system is reliable when its free from failure and when prediction analysis, trend analysis, reliability measurement, defect classification, field data analysis, software metrics, software testing, reliability fault tolerance, fault trees analysis, software reliability simulation and performance metrics are carried out. An analysis of both functional requirement and non-functional requirement is also very important; functional requirement specify recovery features, system-failures protection and error checking while non-functional requirement specify the availability and reliability of the operating system. Unlike customized software, operating systems are developed and used by different customers all over the world. Therefore, a more proper approach is to analyze operating system in terms of systematic quality assurance process, reliability feedback date and quantitative measurement collectively.

The aim of this research is to analyze the reliability of operating systems, our objectives is to obtain reliable operating system using these three approaches in order to have customer satisfaction and confidence. To archive this, we analyzed the reliability of operating systems in terms systematic quality assurance process, quantitative measurement and reliability feedback data collectively. We realized that when these three approaches are used collectively to analyze reliability of operating system. It will improve the quality, dependability, reliability, usability, confidentiality, performance and durability. But due to privacy concern, Mass Market Software (MMS) developers like Microsoft Windows do not reveal all their details with academic research group. Therefore, only few are made available. In this paper, we reviewed the reliability analysis of operating systems. Section II review related work. The research methodology is explained in section III. Section IV discussed conceptual frame work and data collection.

II. RELATEDWORK

Some works have recently been done on software reliability of mass market such as assessments of mass market software [13]. Software developers require data of failure that affect reliability experience, other approaches such as testing data and user's satisfaction survey data are not sufficient enough to analyzed operating system. Specific failure is identified using testing data approach but does not show impact on user's reliability experience. Information on user's reliability experience can be provided by user's satisfaction survey approach but this approach does not address failure that software developers can work on. Another approach that is used to collect reliability feedback data is web based bug reporting tools and call centers in which users self-report failures. However this approach has many draw back [3].

Some researchers focused on architectural and statistical assessment of reliability of operating system while others focused on modeling and designing tools that can assess the reliability of an operating system. Web based tool for software reliability modeling in which architectural issue and technical decision involved in pointing a standalone tool [10]. Pankajjalote discussed issue in determining reliability of commercial software product. Difficulties in applying architectural software reliability analysis are required. [7].

It is easier to analyze the reliability of Software alone than both hardware and software as a whole and it is no longer possible to test all possible combinations of user configurations for instance in Windows XP there are presently more than 35000 drivers with each driver having more than three versions making the hardware and drivers for all practical purpose infinite. Moreover, it is virtually impossible to capture the usage profile of the product [3].

III. EVALUATION OF MODELS FOR ANALYZING RELIABILITY OF OPERATING SYSTEMS

We present three models used for analyzing the reliability of operating systems; systematic quality assurance process, quantitative measurement and reliability feedback data. Unfortunately, due to privacy concern most operating system company do not make all their information available for research purpose, only little is readily share with academic research.

A. *Systematic Quality Assurance Processes*

Systematic quality assurance process of reliability analysis is carried out by predicting and preventing the error before its occurrence and ensures it covers the entire package. It is a set of reliability analysis technique designed to ensure that the operating system is reliable and maintainable. It increase the confidence of customer, increase the credibility of company, improve work processes, increase efficiency and enable company compete with others. Systematic Q/A process ensure that control of the product is maintained at each stage in the development of the operating system, and the quality is analyzed by quality assurance department. Systematic quality assurance processes emphasize on identifying the defect before they get into product. Preventive action are been taken to eliminate the defects in the product and will be involved in the development and final packaging of operating system. Defect present and defect that may arise in future are analyze and fix.

Defect Analysis

Software defect is the most common cause of customer outage; therefore reliable and quality software is an asset for an organization. One of the disadvantages of software defect is that it is not fully understood to provide means of correcting or avoiding the defect[9]. To analyze operating system defect, we:

- Define the type of errors in order to distinguish programming errors that may make the OS to fail.
- Classify and define the event that cause error to occur

There are two types of errors; regular and overlay error.

Regular Error

Regular error are errors encountered in the field. The failures indication for this kind of error falls into the following categories:

Endless Wait: Continues process without an output.

ABEND (Abnormal End): is an abnormal termination of software.

Addressing Error: This occurs when the OS when incorrect or bad address are used.

Loop: This is when OS goes into continues looping.

Message: An error message is displayed on screen when operating system tries to perform request function.

Overlay Error: Over lay error compose of software errors that resulted from storage overlay. Overlay defect have higher impact on software than regular defect. Overlay often occur within operating system and remain for a long period of time in operating system. How can such defect be corrected? Defects are identify, analyze, correct and thoroughly tested before packaging the software product.

IV. DATACOLLECTION

Due to privacy concern Mass Market S/W developers cannot share all their data with research institution, for this reason, only little is readily share with academic research. Various approaches were used in collecting data to analyze the reliability of windows operating system as discussed in section III. Such approach is required for collecting failure data especially failures occurring from users around the world. Consider the following data collected using CEIP approach:

Table 1a: Reliability Analysis of some Windows OS Family

Versions	Release Date	Known Vulnerability	Known Defect	Known Defect Density
WIN 95	95	50	500	0.3333
WIN 98	98	84	10000	0.5556
WIN XP	2001	125	106500	2.6625
WIN NT 4.0	1996	180	10000	0.625
WIN 2000	2000	204	63000	1.80

Several sources (Operating System data, 2004; MITRE Corporation, 2005; Rodriguez, 2001; national Vulnerability Database, 2005; McGraw, 2003;). The known vulnerability of Windows 95 and Windows 98 are 50 and 84 respectively (per thousand line of code) Windows XP had a higher value of 125 because the available data are version of beta data. It is believed that the version released had less defect, Windows 2000 and Windows NT with 204 and 180 respectively. From the next column, known defect of Windows 95 and Windows 98 are 0.33 and 0.55 respectively, Windows XP with a higher defect of 2.66 because data used were for beta version, but with a fewer defect in the release version, Windows 2000 and Windows NT with defect density of 1.8 and 0.6 respectively. These can be demonstrated by below figure:

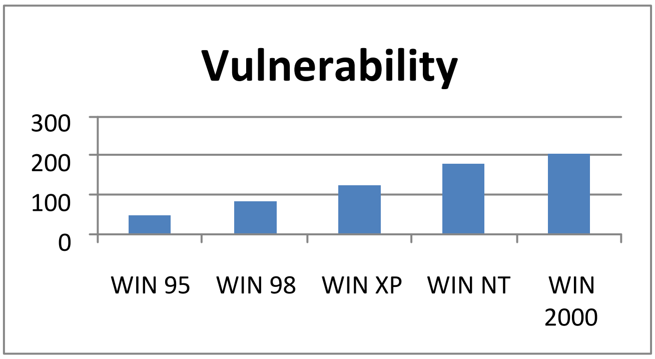

Figure 1: Vulnerability of Windows family

Comparing this with the data of Red Hat Linux operating system in table 1b, we observed that vulnerability and defect density of Linux 6.2 and 7.2 are higher than that of Windows operating system as shown in table 1b.

Table 1b: Reliability Analysis of Linux Family

Versions	Release Date	Known Vulnerability	Known Defect
R H Linux 6.2	Mar 2000	118	2096
R H Linux 7.1	Apr 2001	164	3779
R H Linux 7.3	May 1995	106	1945
R H Linux 8	Sep 2002	86	1824

Table 1b shows values of vulnerability and defect of Red Hat Linux operating system, the known value of Linux 6.2 and 7.1 are 118 and 164 respectively. Comparing these values with that of Windows OS family (Windows 95 and 98 in particular) its observed that Linux has a higher value vulnerability than Windows OS. However, these alone should not be used in comparing the two competing OS versions. These can be represented by below graph:

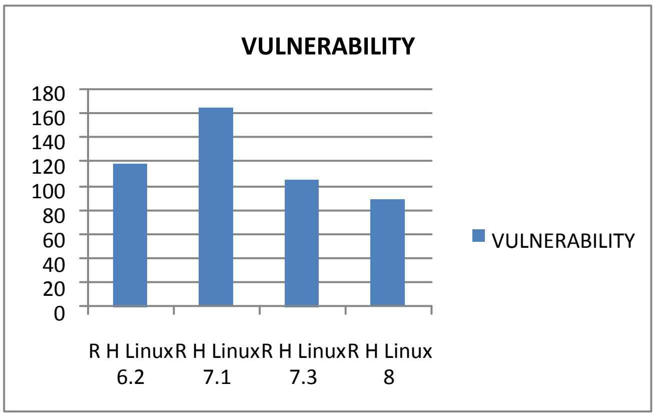

Figure 2: Vulnerability rate of Linux Family

A. *Quantitative Measurement*

Quantitative measurement is a measurement that provide failure free window, quantitative measurement is also use in analyzing the reliability of software and is usually perform by Windows reliability team. Windows reliability team measured rate of failure [13] before the release of the product. Metric are been evaluated across time by reliability team. Several users are required to test the beta-to-release version before it will be released into the market. Consider the beta-to-release of Windows vista analysis as demonstrated by the figure below:

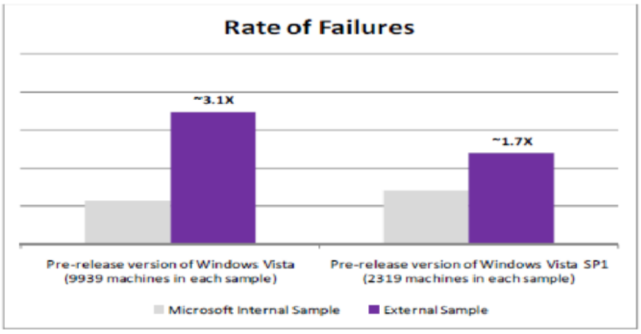

Figure 3: Failure of rates of Microsoft External and Internal machine 9939 (Paul, 2008)

If the total number of failures for N installation, T time with F failure, to estimate the rate of failure of the s/w we use:

$$\lambda = F/(N*T)$$

The reliability growth models uses data collected during testing to predict the future reliability of the product assuming:

- Configuration of the test system are representative of the user environment
- Product usage and management does not impact reliability
- Failure occur once and then corrected

B. Reliability Feedback Data

Information that identifies occurrence of failure is called reliability feedback data. The occurrence of failure; such failures include application crash, application hangs and operating system crash. There are four method of collecting feedback data. These include:

1. Online user reporting: Feedback of failure is sent using website or feedback application. The user complete the predefine form (to provide the feedback data) before sending it. Operating system developers contact users for more information about the failure.
2. Interactive user reporting: user contact live operator or the producer and interact to report every failure, the live operator may then collect the feedback information and complete the predefine form. The user may later be contacted for additional information.
3. Automated pre-incidence reporting: Windows automatically initiate a feedback when it detects error or when failure occurs. Windows automatically collect the feedback data and send to windows developer (Producer). Example of such error is windows error reporting.
4. Automated reliability monitoring: System automatically sends reliability feedback to producer (Microsoft) after an initial consent. An example of Windows reliability monitoring is Windows reliability analysis component.

C.Reliability Analysis Component (RAC) A new feature designed to provide timely and accurate feedback data in Operating System such as Windows is called Reliability Analysis Component (RAC).. RAC collect data from user opted into Windows customer experience improvement program (CEIP), RAC is a light weight, secure and user privacy complaint ensuring user trust is a key consideration [13]. RAC runs on background (as low priority process) and gathers data using event log and system calls. RAC regularly sends data (feedback data) to Microsoft through a secured connection. Presently, RAC tracks hundreds of thousands of real world windows vista machine and collect data from CEIP and all the failures are recorded and corrected. RAC has advantage of continues monitoring of Windows OS instead of reporting after some period of time.

Customer Experience Improvement Program (CEIP)

Customer Experience Improvement Program (CEIP) is an elaborate, event recording and programmable product that can be use to record the usage and failure data. Microsoft office (Windows 2003) was the first system that CEIP was applied to. In order to record failure of Windows via CEIP, the product must be programmed to record event. Three types of event are captured:

- Assert Failure: an application shipped to customer contain assert statement called ship assert. Failure is ascertain when the ship assert fail, it is then treated as an event and the information about the failure (failing assert) is recorded.
- Alert: when special situation arise, user receive alert from most applications. For instance alert message is displayed if file writing fails, copying file fails and if file does not exist.
- Application termination: An event is recorded when an application terminates. Normal exit, hangs, crash exit and user forced exit are all recorded. Some of the event may not be recorded until the application restart.

Application termination event are mainly used in identifying crashes and hangs, alert event and asset are used in identifying the functionality failures. Each failure (for both alert and assert) is recorded as separate event and post processing of event log categories separated the event into groups. Whenever the event is collected, configuration information such as language been used, patches uploaded, amount of memory in the system and version of the underlying operating system will also be collected. For this reason, other updates and loading of patches can be identified. CEIP is made up of two component; a centralize server and clients (residing in user system). The servers receive configuration data and the recorded data from the client system. Parameters to record are been specify by the event logging mechanism, such parameters include unique user tag, application name, program counter and alert. These parameters are responsible for defining different bucket, method of bucketing assist in identifying the major causes of failure and event. If a number of problems are responsible for the main causes of failures encountered by users, the problem will be given high priority issue to resolve by windows developer in order to give maximum reliability.

CEIP reports are used by Microsoft to report quality metrics and reliability because it is very effective in reliability analysis of windows operating system.

Table 2: An example of CEIP report

Widows Versions	No. of Session	Session Length (min.)	No. of Crash Failure	Crash Failure Rate (Per Hour)	No. of Hang Failure	Hang Failure Rate (per min)
WIN 2000	3300	3,140,00	30	0.57	100	0.191
WIN XP	200	254000	10	0.023	70	0.165
WIN VISTA	1200	0000	10	0.066	20	0.133
WIN 7	648000	260	290	0.080000006	29.5	0.0000003

Application termination event are mainly used in identifying crashes and hangs, alert event and asset are used in identifying the functionality failures. Each failure (for both alert and assert) is recorded as separate event and post-processing of event log categories separated the event into groups. Whenever the event data is collected, configuration information like language been used, patches uploaded, amount of memory in the system and version of the underlying operating system will also be collected. For this reason, other updates and loading of patches can be identified. If a number of problems are responsible for the main causes of failures encountered by users, this problem will be given high priority issue to resolve by windows developer in order to give maximum reliability. CEIP reports are used by Microsoft to report quality metrics and reliability because it is very effective in reliability analysis of windows operating system.

While CEIP is used for recording failure data, Microsoft Corporate Error Reporting Software is used for collecting crash data. It directly receives crash report using a configured server with shared directory.

Microsoft Reliability Analysis Service (MRAS)

Unlike Customer Experience Improvement Program (CEIP), Microsoft Reliability Analysis Service emphasizes on the reliability analysis of Windows servers such as Microsoft SQL database, Windows Active Directory (WAD), MS Mail Exchange and MS IIS web server. Recall that in CEIP data is reported directly to only Microsoft, custom reliability reports for server are provided to customers in MRAS. The two main components of MRAS are MRAS reporting site and MRAS client.

MRAS Reporting Site: MRAS reporting site serve as the warehouse and analysis component [12] where the data from client is been analyzed, loaded and stored in tables for subsequent reporting. Web interface is used to access and managed the reliability report.

MRAS Client: In MRAS client, a set of servers for tracking has to be supplied to the MRAS client and the MRAS client is installed on a particular server. One of the functions of MRAS client is the collection of server log data and subsequently uploading it to the MRAS reporting site. Client MRAS collect only new data in each collection which make frequent collection efficient. For instance, if there are hundreds of events to be recorded in a servers log, only a relatively small number (100 in different event on the approximate) will be collected. The server group specification are been monitored by the user which makes the client to number the servers is monitoring. It also obtain server configuration data which application are running on it. MRAS focused mainly on crashes and event leading to shut down of the system or application. Break down of known shut down reasons are been provided by addressing and understanding the failure and cause of down time force to shut down which is a feature in Windows server.[3].

Apart from shut down information, information on crashing application modules and other information that are relevant for reliability analysis are also been provided by MRAS. More than 200 corporate users deployed with MRAS beta version and are extensively been used by OS developers such as Microsoft to connect thousands of servers. MRAS beta versions are also used for reliability analysis on Windows 2003.

Criteria for Evaluating Feedback Data

The criteria below are important in windows reliability analysis so as to meet customer's expectation:

- Correct: Windows required accurate data to properly address failure, an omission or mistake while entering data can lead to lack of solution to such failures [3].
- Accurate: All information in feedback should be accurate so that maximum improvement in reliability can be achieved. Windows developer focuses in correcting all failures from the feedback data received.
- Comprehensive: All failure needs to be reported in feedback, Windows developer cannot weight failure appropriately if feedback is not comprehensive enough.

Table 3: Analysis of Error Types and Their Defect (Archana, 2009)

Type of Defect	IPL	Percent of APARS	HIPERS
Data Error	10	2	14
Synchronization	0	0	0
Register Reused	0	12	0
Pointer Management	0	6	0
Allocation Management	0	4	11
Uninitialized Pointer	0	6	0
Undefined State	0	10	73
Statement Logic	0	4	0
Copying Management	0	4	0
PTF Compilation	0	22	0
Sequence Error	0	4	0
Unknown	0	4	0
Other	0	5	0
Address error	16	38	37
Error Message	0	4	2
ABEND	21	32	28
Endless wait	21	4	5
Incorrect	16	4	13
Infinite Loop	21	4	17

A significant observation from table 3 shows that more than 90 percent of the errors are detected and corrected after reliability analyses which suggest that the operating system comes across code containing errors before failing. This kind of failures causes process to wait endlessly for an event that will never happen. Such errors are basically different from hardware errors. Just like in the design of hard ware in which fixed errors never reappear, for this reason a systematic and quantitative approach are used to analyze and correct these errors before packaging.

Table 4: Analysis of crashes cause by application (Patterson, 2005)

Application Categories	Usage %	Crash%	No. of Crash
Database	N/A	1%	8
Email	24%	8%	119
Instant Messaging	N/A	1%	17
Security	N/A	1%	9
Mult1media	6%	4%	32
Web Browsing	18%	38%	562
Remote Connector	N/A	2%	27
Scientific Computing	7%	6%	91
Code Development	10%	2%	27
Input-Output	N/A	1%	16
System Management	4%	1%	8
Document Viewer	8%	6%	84
Document Archiving	N/A	2%	32
Document Preparation	22%	11%	155
Unknown	N/A	17%	247
Other	1%	1%	15

Table 4 shows the distributions of crash during reliability analysis of Windows XP, application crash are more frequently than Windows operating system crash. Out of 1546 crashes only 72 are caused by OS and the remaining 1474 are caused by application and 55 by Windows explorer. Different drivers (operating with kernel-level) are the causes of the remaining 17 crash. Windows crash is very frustrating than application crash, Windows crash include blue screen generating crash. How then can Windows be more reliable in spite of application crash? Such events are systematically fixed in such a way restarting the crash application. Web browser causes most of the crash in the data set (Internet explorer as an instance). Plug-ins running inside the browser are the main causes of large number of crashes in the browsers but the analysis tool always blame the browser instead of the plug-in. One other application crash is the document preparation software (MS-Word, Outlook and Power Point).

While most of the applications listed in table 3 are causes of crash, it is unfair to justify the reliability of these applications.

One of the reasons that Windows OS is more reliable is because all errors identified are analyze and corrected before any testing activities. A fair reliability of Windows OS require statistical usage and analysis of the other data which assist in identifying data skew. For each machine, it is very useful to know and get record for performance metrics prior to crash, during the crash and after the crash, this will make it easier to analyze the causes of the crash and fixed. For instance, some of the causes that lead to Windows crash are the amount of free space, system uptime and processor queue length. All these can assist in suggesting the sequence of event that lead to crash and the processes as well as factors that influence the progression of the failure.

However, one of the approach used to analyze this more accurately is by collecting machine metrics and process information but due to legal reasons, Microsoft do not make all these available for our research. Several other limitations are imposed on this analysis due to privacy by Microsoft. In order to analyze windows ideally, we will need to know the specific duration taken by each application and the resources consumed which is absent in data collected.

V. CONCLUSSION

In this research, we have discussed three methods of analyzing the reliability of operating systems and it has been observed that Software will be more reliable if these three methods are used collectively. To meet the quality expectation of customers, systematic quality assurance process is used. It provide a means of predicting and preventing errors before it occurrence and ensures it covers the overall package prior to release. The quantitative measurement is carried out by Windows reliability team to provide failure free Window. The team evaluates metrics across time and run time.

The last insight is by the use of feedback data, which include information that identifies the occurrence of failure. Such failures include application crash, application hangs and operating system crash. The crash data collection has contributed to analysis of Windows reliability, it can reveal that Windows is very reliable and Windows OS is not responsible for most of the personal computer crashes. Applications software is responsible for most of these crashes, especially browsers. Customer Experience Improvement Programmed (CEIP) which collects the usage data and detailed failure through logging of different events was also discussed. We also describe Microsoft Reliability Service (MRAS), where products running on server record their own event. MRAS uses logging mechanism of Windows server. The event log is then sent to central place, it is then analyze and report is given. One of the advantages of this approach is that the size of the observed group is known as only specified servers are monitored.

Reliability Analysis Component (RAC) which is a new feature in Windows Vista, it provide accurate and timely reliability feedback. RAC collect data from user opted into CEIP. Both CEIP and MRAS provide detailed information for analyzing the reliability of Windows OS in order to improve the quality and improve customer expectation.

A proper publicize reliability analysis of Widows OS enable users to guide their usage patterns and purchasing decisions. Microsoft is striving to provide customers with a better reliability experience.

REFERENCES

[1] Archana Ganapath and David Patterson (2009). *Crash Data Collection: A Windows case Study*. Retrieved November 15, 2012 from http://citeseerx.ist.psu.edu/viewdoc/summary?

[2] Buckley, M., and Chillarege, R. (1995). *Discovering Relationship between Service and customerSatisfaction*. (192-201).

[3] Brenden Murphy and Mario R., Garzia (2004) *Software Reliability Engineering for Mass-Market Product.* Microsoft Corporation 8(1).

[4] Charles, and Mario Garzia (2007). *Making Windows Vista Reliable: An Introduction to Windows Reliability*. Retrieved November 15, 2012 from http://channel9vip.orcweb.com/showpost.aspx?postid=28693k

[5] Customer Experience Improvement Program. Retrieved January 20, 2013 from http://www.microsoft.com/windowsvista/privacy/vistartm.mspx

[6] Garzia, M., Khambatti M., and Ni, M., (2007). Assessing End User Reliability Prior to Product Ship: *Reliability Analysis of System Failure Data.* Retrieved December 5, 2012 from

http://www.deeds.informatik.tudarmstadt.de/RAf07/programhtml

[7]Heiko Koziolek, bastian schlich and Carlos Bilich (2010). A Large Scale Industrial Case Study On Architecture-Based Software Reliability Analysis. Retrieved November 15, 2012 www.informatik.uni-trier.de/~ley/pers/.../**Schlich**:**Bastian**.html

[8]Introducing Windows Error Reporting. Retrieved from http://msdn2.microsoft.com/en-us/isv/ibb190483-asp

[9]Mark Sullivan, Ram Chillarege.(1991). *Software Defect and their Impact on System Availability:A Case Study of Field Failures in Operating System.* Retrieved November15,2012fromhttp://www.chillarege.com/articles/software-defects-impact-mvsapar

[10]Michael Lyu R.and Jurgen Schonwaldev (1998).WEBCASRE: *A Web-Based Tool for SoftwareReliability Modelling.* Retrieved December 7, 2012 from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.46.4248

[11]Musa J. D., Lannino and Okomotok (1987) Software Reliability Measurement, Prediction and Application. Retrieved November 18 from http://scindeks.ceon.rs/article.aspx?artid=145071961003046K

[12] Pankaj Jalote, Brendan Murphy, Mario Garzia, Ben Errez (2004). Measuring Reliability of Software Products. Retrieved November 28, 2012 from research.microsoft.com/pubs/70136/tr-**2004**-145.pdf

[13] Paul Luoli, mingtian Ni, Song Xue, Joseph P. Mullally, Mario Garzia, and Mujtaba Khammbati.(2008). Reliability Assessment of Mass Market Software: Insight from Windows Vista.Retrieved December 18, 2012from:http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4700332&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5 %2F4700288%2F4700289%2F04700332.pdf%3Farnumber%3D4700332

[14] Patterson, O. (2005). Crash Data Collection on Windows Case Study. International journal on Dependable System and network