Materials Science Research india Vol. **4**(1), 225-229 (2007)

An wholistic approach to web applications maintenance

PRIYANKA TRIPATHI, M. KUMAR, and N. SHRIVASTAVA

[1]Department of Computer Application,
Maulana Azad National Institute of Technology, Bhopal - 462 023 (India)

(Received: May 20, 2007; Accepted: June 26, 2007)

ABSTRACT

It is a well-established fact that the Web applications require frequent maintenance because of cutting–edge business competitions. Further, such applications are maintained by third party in majority of the cases. This paper emphasizes that the factors like Analyzability, Changeability, Scalability, Stability and Testability as defined by ISO9126 standards are not sufficient. It is important to consider factors like the capability of maintenance team, management attitude of both client and maintenance organization. Further, the capability of the original Web application developer is also important.

Key Words: Web applications, Quality, Maintainability factors, Maintainability sub-factors.

INTRODUCTION

Web applications are different from traditional software systems in the sense that they involve heterogeneous technologies in hardware as well as software. For successful development of large Web applications , we need a team of people with wide ranging knowledge and skills. We need Graphic designers to develop the look and feel, we need people with library science background to organize, navigate and search information. We need database designers and programmers to develop code, network security and other security aspects. We often involve architects to get better aesthetics in the Web applications. The code development will involve hypertext structures, JSP, Servlet , scripting languages, etc.

Another difference is the difference in the unit of measurement of a metric for each application domain. For traditional systems, the unit of measurement of a metric can be a file, procedure or an attribute. For object-oriented systems, the unit of measurement can be a class, interface or attributes. For web applications, the unit of measurement is a web object which can be either an HTML file, JSP, Servlet or client scripts. Hence models and metrics for traditional systems can not be applied to web applications[1].

Most Web applications involve critical business assets which promote their services through internet. Because of globalization and cut-throat business competition, these Web applications evolve continuously during their life-cycle.

Lehman *et al.*[2] gave two laws of software evolution that affect the evolution of Web applications. They are

1. The law of continuing change: A program used in real world must change or eventually it will become less useful in the changing world.

2. The law of increasing complexity: As a program evolves it becomes more complex and extra resources are needed to preserve and simplify its structure.

An example for Internet evolution is amazon.com a leading e-commerce web

application. Amazon.com started with 0 customers in 1995. In 2003 it had around 20 million customers and the largest online store in 220 countries[3].

It is a common practice that Web applications are hosted and maintained by third party. Because of heterogeneousity of such Web applications, the maintenance becomes a cumbersome process and becomes impossible to predict maintenance cost using traditional models and metrics. In this paper, the authors have introduced some factors that play important role in the dynamics of maintenance , especially when it is a third-party maintenance.

Software maintainability

Maintainability is an important attribute in all the software applications as it is learned that only 25% to 33% of the total effort put in during the complete life cycle of a software system goes in actually building the system [4,5,6,7]. The rest is consumed by effort expended towards the operational maintenance of this system. This clearly indicates that maintenance takes more efforts as compared to the development of the software.

The software quality and maintainability are directly related[5], i. e. a good quality software is expected to have low maintainability. The authors[8] in a recent paper have discussed the quality attributes of Web applications and mentioned that maintainability is also a quality factor. Pressman[9] describes maintainability as the ease with which a program can be corrected if an error encountered, adapted if its environment changes, or enhanced if the customer desires to change the requirements.

The ISO9126[10] standards provide a hierarchical structure of factors and sub-factors, see Fig 1 of maintainability. These factors when accounted in the software design give rise to a good quality software . In other words if the software design takes care of six main factors namely analyzability, changeability, scalability,stability and testability, it will be simpler to maintain.

The software maintenance as defined in IEEE standards[11] is: The modification of a software product after delivery to correct faults, to improve performance or other attributes or to adapt the product to a modified environment. According to Basili and Mills[12] the software maintenance may be looked as :

Most software systems are complex, and modification requires a deep understanding of the functional and non-functional requirements, the mapping of functions to system components and the interaction of components.

According to Lienz[13] and Swanson[14], software maintenance can be categorized as corrective, adaptive, perfective and preventive. Traditional maintenance means restoring something to its original shape, software maintenance deals with fixing problems in original system, and bridging the gaps between the operational system and the specifications. With end-users using the software, there is always scope for improvements in the operational system, or technical improvements. Changes to the operational environment software as well as hardware platform also make it necessary to make appropriate changes to operational software. We also need to distinguish between maintenance changes made to software in a development environment as opposed to those made while the software is in operation.

The maintainability of software system has always been a problem with software professionals. Since the third-party maintenance is now becoming a reality as more and more organizations are opting for third-party maintenance of their Web applications. It is the high time that software maintenance be looked in the right perspective so that a realistic cost estimates be prepared for the software maintenance. The authors asserts that realistic cost estimates are only possible with the help of suitable model and metrics. This is described in the next section.

We further assert that in view of third-party maintenance of Web applications, there are some more factors that contribute to maintainability. It is therefore we propose to incorporate following factors that are more realistic and useful in the hierarchical structure of Fig. -1.

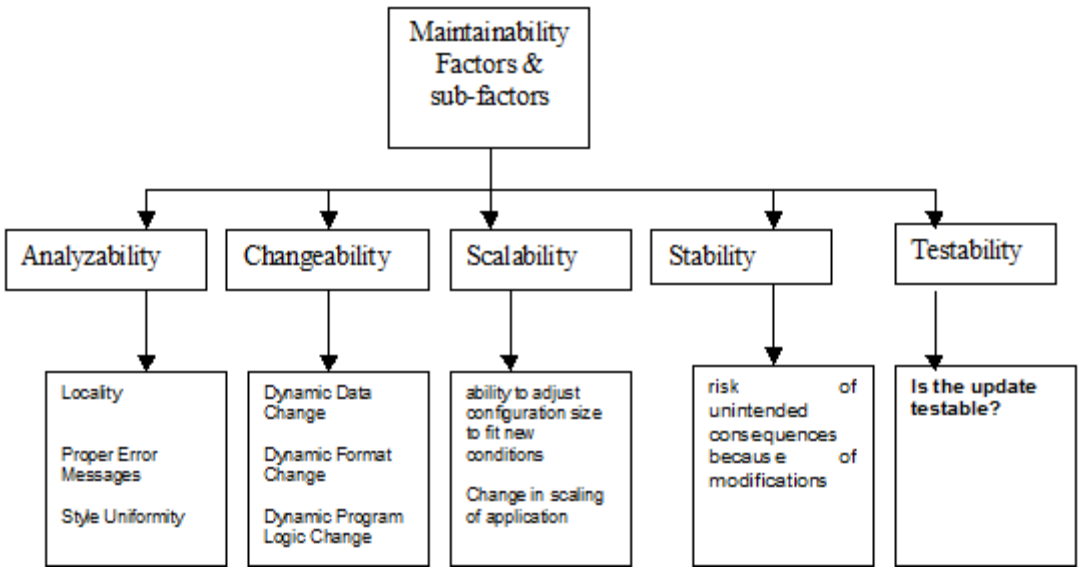

Fig. -1: Factors & sub-factors of Maintainability described by ISO9126

Complexity of Web-Application

This factor plays an important role in maintenance of the Web applications. Depending upon the complexity of the Web applications , it should be estimated well in advance that how critical is the maintainability. Banker, Kemerer et. al.[15] found that complexity does make software harder and more costly to maintain but did not examine the notion of increased complexity with each modification.

Development Methodology

Be it an evolutionary model or agile development of the web-application, it should consider the maintainability aspect of the software right in the beginning of the software. In case of agile development it is more difficult to perform the third party maintenance because in most of the cases to speed up the process no Software Engineering Rules are followed.

Programmer's Skill

Programmers who have reasonable experience in maintaining a Web-Application are very few. Maintenance is yet not given its due importance over development. Perhaps this is the reason people don't want to do this and want to be in development of the software. In case of a third party maintenance it is very important that the team members have reasonable experience because the program is not designed and developed by their team and by someone else's. There can be a difference of opinion in the program logic.

Standards of Coding

This is a factor which influences the process very much. If the coding standards have been followed it is very convenient to read the code and understand the logic. In contrast to that if the coding standards have not been followed it becomes very difficult to incorporate a small change even. This discipline should be exercised by the development team for the convenience of a third-party maintenance

Documentation Completeness

It is a must for the people who are maintaining the system to have full details and documents of the program when it comes to third party maintenance. Without proper documents it is very difficult and sometimes impossible to maintain the web-application.

Management Attitude

If management is supportive and keeps a positive attitude for maintenance, the performance

of the team will be better which will eventually earn profit for the organization.

Customer Attitude

It is very important especially in case of third party maintenance that the customer should be considerate towards the maintenance process. Because the web-application is designed and developed by the other party , it might take some time to analyse and incorporate the change request. It is important to know that the web-sites can have maintenance cycle of days or even hours

Hierarchical maintenance model

In view of issues discussed in Section 3 , we propose following 12 factors and 26 sub-factors that contribute for calculation of maintainability index.

Analysability

1. Locality
2. Average number of components per task
3. Style uniformity

Changeability

1. Dynamic data change
2. Dynamic format change
3. Dynamic program logic change

Scalability

1. Ability to adjust configuration size to fit new conditions
2. Change in scaling of application

Stability

1. Risk of unintended consequences because of modifications

Testability

1. Is the update testable?

Complexity of Web-Application

1. Complexity of the system (High, Medium, Low)
2. Criticality of the system

Development Methodology

1. Agile development/ Sufficient time has been given
2. Is it a third-party maintenance?

Programmer's Skill

1. Experience of the programmer particularly in maintenance
2. Does the programmer wish to continue in the same project or wants to switchover to other development project.
3. Persons involved in maintenance were also involved in development.

Standards of Coding

1. Coding standards have been followed or not.

Documentation Completeness

1. Documents at the initial stage
2. Documents at completion of the project
3. Completeness of Design documents, Test cases and Help manual

Management Attitude

1. How much interest does the management take in maintenance?
2. Stability of the management

Customer Attitude.

1. Customer's IT Team / IT literacy
2. Interaction between the end user and the customer
3. Interaction between the developer and the customer

Quantitative metrics and models for predicting web applications' maintainability can be of a great help in order to control the maintenance cost of web applications.Thus we can define a Maintainability Index (MI) for measuring efforts of maintainability **MI = F($w_i f_i$) , i= 1 to 12**

where, w_i are the weights , f_i are the sub-factors , and F is the additive function.

Conclusion

This research has identified main factors and sub-factors that affect overall maintainability of Web applications. It should be noted that the attitude of programmers, customers and the management plays a crucial role in the maintenance of Web applications. We are in the process of evolving weights for each factor and sub-factors , based on our controlled experiments , so that the maintainability index (M**I**) can be calculated precisely.

REFERENCES

1. Emad Ghosheh , Jihad Qaddour , Matthew Kuofie and Sue Black, *A Comparative Analysis of Maintainability Approaches for Web Applications*, **1155 1-4244-0212-3/06 ,IEEE** pp 1155-1158 (2006).
2. M. Lehman, J. Ramil, P. Wenric, D. Perry and W. Tursky, *" Metrics and laws of software evolution the nineties view."* In proceedings of the 4th International Software Metrics Symposium, **IEEE Computer Society Press**, pp. 20-32(1997).
3. Len Bass, Paul Clements, and Rick Kazman, *Software Architecture in Practice*, Addison-Wesley, **2 edition**, 2003.
4. Zelkowitz, Marvin V., *Perspectives in Software Engineering*, ACM Computing Surveys (CSUR) archive, **10**, issue 2, pp197-216 June (1978).
5. B. Lilburne, P. Devkota, K. M. Khan, *Measuring Quality Metrices or Web applications*,IRMA International Conference , New Orleans, USA, (2004).
6. Arun K. Mishra ,Pankaj Bhatt, *"Influencing Factors in Outsourced Software Maintenance"* , SIGSOFT Software Engineering Notes Page 3, **31**; **3**,May (2006).
7. Arun K. Mishra, Pankaj Bhatt, *"Dynamics of Software Maintenance"* , ACM SIGSOFT Software Engineering Notes, **29** Number 5,Page 1 September (2004).
8. Priyanka Tripathi, M. Kumar, *" Some Observations on Quality Models of Web Applications"*, International Conference Of Web Applications icwa2006, Bhubaneswar , Orissa, India, 23-24 December (2006).
9. Pressman R. S. , Software Engineering A Practitioner's Approach, 6th edition, McGraw-Hill International Edition (2005).
10. ISO/IEC FDIS 9126-1 : Software Engineering – Product Quality Model (2000).
11. IEEE Std. 1219: Standard for Software Maintenance, IEEE Computer Society Press, (1993).
12. V. Basili, and Mills, *" Understanding and Documenting Programs."* IEEE Transactions on Software Engineering SE-8, 270-283 (1982).
13. B. P. Lientz, and E. B.Swanson, *Software Maintenance Management*, Addison Wesley, Reading MA, (1980).
14. E. BurtonSwanson, *The dimensions of maintenance*, Proceedings of the 2nd international conference on Software engineering, San Francisco, California, United States, pp492-497 (1976).
15. Banker, D. Rajiv , Datar, M Srikant , Kemerer, Chris F., and Zweig , Dani., *Software complexity and maintenance costs. Communications of the ACM*, **36**, 81-94(11 Nov 1993).