IOSR Journal of Computer Engineering (IOSRJCE)
ISSN: 2278-0661, ISBN: 2278-8727Volume 7, Issue 4 (Nov. - Dec. 2012), PP 53-59
www.iosrjournals.org

Analyzing the Assessment Criterion for Web-based Distributed Software Development

Dillip Kumar Mahapatra[1], Tanmaya Kumar Das[2], Gurudatta Lenka[3]
[1](*Department of Information Technology, Krupajal Engineering College, Bhubaneswar, Odisha, India / Biju Pattanaik University of Technology, Rourkela, Odiaha, India.*
[2] *(Department of Computer Science & Engineering, C.V. Raman College of Engineering, Bhubaneswar, Odisha, India / Biju Pattanaik University of Technology, Rourkela, Odiaha, India.*
[3] *(Department of Computer Science & Engineering, Krupajal Engineering College, Bhubaneswar, Odisha, India / Biju Pattanaik University of Technology, Rourkela, Odiaha, India.*

Abstract: *Globalization has brought changes to the software development era. The traditional software development processes are greatly incorporated with the distribution policy of software development though WEB based technology and applications. A distributed system is a computing system in which a number of components cooperate by communicating over a network. The explosive growth of the Internet and the World Wide Web in the mid-1990's moved distributed systems beyond their traditional application areas, such as industrial automation, defense, and telecommunication, and into nearly all domains, including e-commerce, financial services, health care, government, and entertainment. This paper analyses the key criterion of developing different software projects through web-based development strategies and evaluates the key factors to resolve its associated challenges, including distributed object computing middleware, component middleware, publish/subscribe and message-oriented middleware and web services.*

Keywords: *Distributed software development, Complexity, Inter-disciplinary collaboration, Success factors, System architecture*

I. Introduction

Computer software was traditionally used in *stand-alone systems*, where the user interface, application 'business' processing, and persistent data resided in one computer, with peripherals attached to it by buses or cables. Few interesting systems, however, are still designed in this way. But, most computer software today runs in *distributed systems*, in which the interactive presentation, application business processing, and data resources reside in loosely-coupled computing nodes and service tiers connected together through the networks. Despite of the increasing ubiquity and importance of distributed systems, developers of software for distributed systems are influenced by number of following factors such as [Ref. 01]:

- *Inherent complexities*, which arise from fundamental domain challenges: e.g., components of a distributed system often reside in separate address spaces on separate nodes, so inter-node communication needs different mechanisms, policies, and protocols than that of the others those are used for intra-node communication in a stand-alone systems. In this respect, synchronization and coordination is more complicated in a distributed system since components may run in parallel and network communication can be asynchronous and non-deterministic. The networks that connect components in distributed systems introduce additional forces, such as latency, jitter, transient failures, and over-load, with corresponding impact on system efficiency, predictability, and availability.
- *Unusual complexities*, which arise from limitations with software tools and development techniques, such as non-portable programming APIs and poor distributed debuggers. Ironically, many accidental complexities stem from deliberate choices made by developers who prefer low-level languages and platforms, such as C and C-based operating system APIs and libraries that scale up poorly when applied to distributed systems. Since, the complexity of application requirements increases, moreover, new layers of distributed infrastructure are conceived and released, not all of which are equally mature or capable, which complicates development, integration, and evolution of working systems.
- *Inadequate methods and techniques*, in analysis and design phases of the software products have focused on constructing single-process, single-threaded applications with 'best-effort' quality of service (QOS) requirements. The development of high-quality distributed systems particularly those with stringent performance requirements, such as video-conferencing or air traffic control systems has been left to the expertise of skilled software architects and engineers. Moreover, it has been hard to gain experience with software techniques for distributed systems without spending much time wrestling with platform-specific details and fixing mistakes by costly trial and error.
- *Continuous re-invention and re-discovery* of core concepts and techniques. The software industry has a long history of recreating incompatible solutions to problems that have already been solved. There are dozens of general-purpose and real-time operating systems that manage the same hardware resources. Similarly, there are dozens of

incompatible operating system encapsulation libraries, virtual machines, and middleware that provide slightly different *APIs* that implement essentially the same features and services. If effort had instead been focused on enhancing a smaller number of solutions, developers of distributed system software would be able to innovate more rapidly by reusing common tools and standard platforms and components.

The Internet and its applications, especially the *World Wide Web(WWW),* have become an integral part of our world and the large number of web users increases day by day. This *Wide Area Network(WAN)* supports the collaboration of distributed project teams and web-based project management. But web-based distributed applications have to be implemented as well.

This paper entertains "what exactly the distributed web-based projects are and how they differ from conventional projects? Usually Web-based projects include creating, changing or expanding a website. A website consists of several web pages which belong to an *URL (Uniform Resource Locator)* and can be accessible either public (Internet-site) or only inside a specific network (Intranet-site). Website are also called portals if the content is very manifold covering many different functional areas. Portals usually also include many features. [Ref. 18].Web projects are the aggregative uses of *IT (Information Technology)* projects and design projects as well.

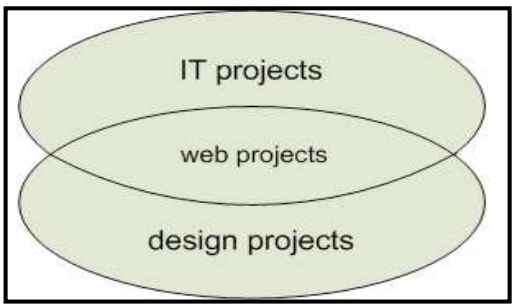

Figure 1: Aggregation of IT and design projects in web- based projects

A web-based distributed software project mainly focuses on the technical aspects, for instance internal changes of data access methods to the used data source without changing anything in the user interface. But generally design-driven web-based projects are also possible like, for example, changing only pictures on a website. Web projects are usually a mix of both sides, making web projects an interdisciplinary challenge [Ref. 07].

II. Assessing Characteristics

There are only a few but very elementary specific characteristics which distinguish web projects from others. These differences result in consequences affecting the whole project management in a distributive way.

2.1 Interdisciplinary collaboration

Distributed web-based projects are inter-disciplinary. The scientists have to work together with designers, business economists, marketing specialists and sometimes even editors and jurists who are responsible for the content of the website. It leads to extended communication effort in the project team and possibly other departments. Interdisciplinary teams which are frequently new assembled pose special challenges to the leadership of the project team. In addition, the workflow of the project is interdisciplinary as well. Clearly sequential workflow like, for instance, *Business to Design and Design to Deployment* is not possible when dealing with distributed web-based projects.

2.2 Cost planning

Cost planning is never a trivial task when a project is planned. But, there exist multiple models to estimate project costs. These projects make this task more difficult because of the mentioned collaboration of design, IT and other departments. There is no established pricing model for such projects. Furthermore, web-based projects may have higher ongoing costs than other completed projects due to content management, usability enhancement and hosting of the website.

2.3 Variant user groups

Distributed web-based projects address very different users. Internet accessible websites are much more than intranet accessible websites. Even if the website is defined to one target group, the users can differ in age, level of education, may possess unequal internet experience and different technical equipment. All these aspects should be considered at the time of deployment of these software projects in their associate environment [Ref. 04].

III. Assessing Success Factors

It has been experienced that a significance percentage of web based software projects suffer from several constraints while under deployment. Especially during the internet hype around 21st century many websites and IT projects were launched which were neither finished nor simply ready for usage. Now-a-days, the methods and web-technologies are much more sophisticated and the time for successful web projects is at hand. After all, it is undoubtedly a matter of opinion when a web project undergoes failure. Many researchers have conducted several analyses on web portals based on two criterion: Are the users of the portal satisfied w.r.t. the implementation effort in due proportion to its benefits [Ref. 04].Distributed web-based projects can suffer from all conventional project problems arising during the implementation such as insufficient communication; important decisions are delayed or not made at all, bad project planning, budget reduction etc. which may be of the following reasons.

- The IT department is involved not until after the design is ready noticing that the design is unworkably.
- Defective content.
- The website misses its economic goals.
- Hacker attacks.
- The website does not fulfill the user-satisfaction resulting in less usage of the website.
- Unexpected user rushes exceeding the performance of the website.

While IT-problems are already recognized during the project, the website is either functioning or not. Design problems often appear after the launching the website, for example if the user acceptance is low. When implementing a distributed web-based project some procedures are helpful like involving the user of the website, keeping project durations short or implementing the most important features at the beginning of the implementation process.

IV. Assessing Cost

The cost factors can be divided into one-time and on-going costs. One time costs are for example developing the concept, implementation of IT and design, project management and the costs for the initial content and the IT and usability tests. The highest costs often arise after the website is launched and utilised.These on-going costs cover investments such as content management, hosting of the website, operating surveillance and further usability testing. Furthermore, essential updates, upgrades and extensions to keep the website practical and secure are also needed. From the customer point of view, the work on the website has just begun when the website is launched.

One of the most underestimated expense factors is *content management*. Although it is not a part of IT-specialists or designers, they have to ensure that new content is easily inputted into the website by the editors. Other important expense factors of web projects are [Ref. 06]:

- Including new technological approaches such as *introducing a new framework for developing dynamic web pages.*
- Usage of a content management system, especially if the IT-specialists are not familiar with it yet.
- When implementing websites, there is often a lack of usability.
- Internationalization.

V. Assessing The System-View

To host a web-based distributed software project, an adequate system-architecture is required which contains several components such as a web server, a firewall, possibly a load balancer, web and application server and a database. A widely accepted form of system-architecture is illustrated in figure:2.

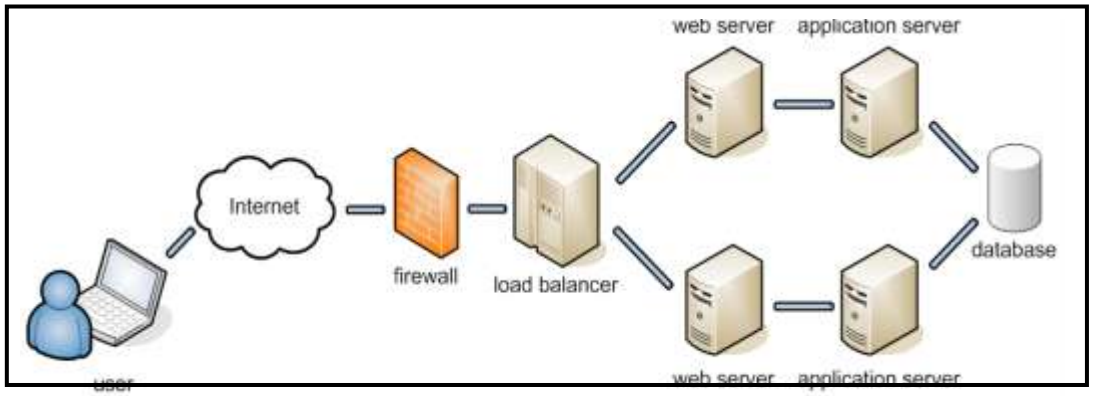

Figure: 2 System architecture of a web-based distributed software project

5.1 Firewall

Usually, an own firewall-computer secures the internal network that it prevents any unauthorized entry. The firewall should be separated from other servers to keep security risks at minimum. The more services are running on a system the merrier security links emerge.

5.2 Load balancer

A load balancer is only needed when huge web-portals are hosted to divide the requests of the clients to several web servers. Whereas load balancing is not very complicated regarding static websites, the sites are simply copied on every web server. Those load balancing systems are sometimes actually integrated in network switches or in small software programs. Handling dynamic portals with user sessions is not trivial. In this case, the user sessions are either closed when switching to another server or have to be mirrored to other servers as well.

5.3 Web server

A web server is a service running on a computer which publishes the web pages of a website via HTTP is known as web server. The process of connecting the host to a web server takes a consecutive course of action:

- A user sends a request from the client via the browser.
- The browser forwards the HTTP-request to the web server which processes it supported by the behind application server and database.
- The web server either sends the requested resources (pages, pictures, etc.) or, in the case of dynamic pages, first generates the according web pages and then sends them via response to the client.

The standard ports are port *80 via HTTP* and port *443 via HTTPS* for encrypted connections. There are many web servers available today. The best known are the *Apache Web server* and the *Microsoft Internet Information server (IIS)* [Ref. 04].

5.4 Application server

An application server contains business logic required by the host computer sharing the resources. The program runs either on the client site of the application server or as a service on the IIS. Application servers govern the logical operations may be too complex or simple irrespective to schema as in J2EE (Java2 Enterprise Edition) framework. Those forms of architecture assure that presentation layer, business logic layer and database layer are separated. An application server also decreases implementation efforts as it includes logic which has to be implemented otherwise separately such as scalability, encapsulation of data sources, logging abilities, etc.

There are no common guidelines from which point of complexity an application server should be used. By all means, the utilization of a commercial application server increases project budgets enormously. Therefore, free available servers are a good alternative for middle-sized web projects which need an application server but are yet too small for investing money for a commercial server. Few examples are the commercial IBM Web Sphere server and the J-Boss server which is available as a free service package.

5.5 Data Server

In all website data has to be stored, and in most cases a data-servers are widely used. Further, there are many options for choosing database i.e.Commercially available or free ones. In this respect Oracle is available with its commercial products, but there are free available alternatives such as *My-SQL* or *Postgre-SQL* fulfilling the required features.

VI. Assessing Via Social Networks

A combination of non-uniform web-based software development and making e-mail communication possible would be the implementation of web based email software. Web-mail facilitates the websites providing access to email functions via a web browser. Many commercial Internet Service Provider(ISPs) offer web-mail as a supplement to their service. But there are also many social networks which allow creating email accounts at no charge like *hotmail, GMX* or *yahoo* and so many social net-works. Free mail-providers usually also support access via *POP3, IMAP* and *SMTP* for using e-mail accounts with an email client program. The greatest advantage of web-mail is instantaneous access to the own email account from every computer connected to the internet and equipped with a web browser. It would be beneficial for project management tools if communication abilities supported by web-mail were included.

VII. Assessing The Usability

Usability is a quality attribute that assesses how practicable an interface is? According to the ISO-norm, usability is the effectiveness, efficiency and the degree of user satisfaction in which specified users achieve defined goals in given environments. A usable program or website is easy to learn, efficient to use and

easy to remember. The word "usability" also refers to methods for improving ease-of-use during the design process. Usability has five quality components [Ref. 21]:

- Learn ability: How easy is it for users to accomplish basic tasks the first time they encounter the design?
- Efficiency: Once users have learned the design, how quickly can they perform tasks?
- Recall capacity: When users return to the design after a period of not using it, how easily can they re-establish proficiency?
- Errors: How many errors do users make, how severe are these errors, and how easily can they recover from errors?
- Satisfaction: How pleasant is using the design?

When implementing web projects, usability should get some additional attention because of multiple conditions. Some of these are [Ref. 21]:

Graphical user interfaces (GUI) have shortened potentials in web projects because of the restrictions which are given by the *World Wide Web(WWW)* and the *Hyper Text Mark-up Language(HTML)* standard. From the technical point of view it is therefore naturally more difficult to achieve the same usability standards for web projects as for conventional Windows programs, for instance .In order to keep navigation through a website clearly laid out and intuitive but nevertheless efficient is one of the most challenging tasks when implementing web projects. In addition, certain browser incompatibilities do not make this task easier.

As the web-based projects are not having bandwidth problems hence, many users of the Internet do not have broadband and need websites which load in acceptable durations. It is observed that no user waits for more than ten seconds when requesting a web page. On the one hand, web projects should provide a sophisticated GUI to compensate web-based limitations, but on the other hand the GUI has to be slim as well to ensure short loading times. These conflictive issues are often not easy to handle.

At the time of implementing a web project, especially in case of commercial website, design and content have to be highly appealing and up to date. On each webpage only the needed content should be displayed. The user must not be confused by amounts of information.

Collection of evaluation report is not a trivial issue in web projects, as web pages are flashed on user`s request. For example, a form which contains many fields should have immediate feedback after entering information, not until the whole form is filled in and submitted by the user. Unfortunately, those features require additional features such as Java-Script. If such features are not possible, detailed and clear feedback has to be given to the user at least after the user submits the requirements.

7.1 Categorization of usability problems

"Severity ratings can be used to allocate the most resources to fix the most serious problems and can also provide a rough estimate of the need for additional usability efforts. If the severity ratings indicate that several disastrous usability problems remain in an interface, it will probably be inadvisable to release it. But one might decide to go ahead with the release of a system with several usability problems if they are all judged as being cosmetic in nature [Ref. 21]".

There is no standardized way for categorizing usability problems. Usually they are divided into critical, major, minor and cosmetic problems [Ref. 21]:

Uncontrolled problems: These problems are imperative to fix prior to the launching of the product. They make the application very difficult to use or even not usable at all e.g. system crashes, workflows break down, complete loss of focus for a specific task, loss of information / data, the user prevents him/her from further use, an obvious competitive feature is missing and *Missing* feedback for critical operations.

Controlled problems: These problems likely to be fixed, hence, normal usage of the product are not affected. They can confuse users while using the product or increase the time a user needs for the performance of a task. Examples for this category are: loss of functionality, problematic impact on a person's workflow, layout inconsistency and non-standard icons for standard functions / a common menu item or tool bar button does not do what is normally expected.

Cosmetic problems These problems need not be fixed unless extra time is available on the project. As already the name points out, they are cosmetic and therefore not affecting the usage of the product in a disturbing way. Examples for this category are: Inconsistent icons (size, color, inappropriate associatively) or links (font, color), Spelling problems and Non-critical workflow issues.

7.2 Usability evaluation

Usability evaluation is a method used to understand how users will use software products. The purpose of a usability evaluation is to predict the expected performance of the actual customer using the current product and materials, as well as detect serious problems prior to the release of the product. Usually, this report provides proposals to improve the quality of the software. Usability testing provides business value and lowers the risk of product failure. For a software product, a low level of usability can increase market risk such as increasing support costs, failing to meet customer usability requirements, loss of brand reputation of a company, or products that are late on the market because of necessary redesign and consecutive loss of market share [Ref. 20].

Usability evaluations should be performed throughout the product development, not only prior to launch. It can be performed on designs, prototypes and final products. It is not necessary to have a complete manual, help system or website to conduct an evaluation. These tests are possible for prototypes, navigation, outlines of manuals or help systems, and specific screens or modules. The product has to be examined to expose both large-scale problems and minor design facts. Therefore, a fairly usability evaluation reduces development time and costs and increases the satisfaction of the consumers [Ref. 20].

As a matter of fact, one of the most important motives for usability testing is that designers or programmers are not the users in the majority of cases. End-users have to be integrated in the whole development process to ensure high usability standards. Project members are routine-blinded if it comes to usability evaluation. It is psychologically nearly impossible to forget or ignore already learned issues when implementing a design or program. Hence, such a think back into the end-users point of view is very difficult for programmers and designers but absolutely necessary for usability testing if no end-users are available. In addition, project team members are usually computer-experienced and use the computer and the Internet quite regularly. They are mostly not the users of the website, except the website is a portal for designers or programmers. In order to evaluate usability of a website and uncover its faults efficiently, the website has to be definitely tested also by the end-users. These test persons should be ideally users of the target group. To quote Nielsen: "Designers are not users" or "The user is always right".

Usability tests can take place for example in a usability laboratory, in a workplace setting, over the web or anywhere else where the required infrastructure is available. There are several options to test usability. Some methods are mentioned below:

*Verification of scenarios***:** The test users have to complete several predefined tasks or scenarios in a specific duration. "A scenario is an encapsulated description of an individual user, using a specific set of computer facilities to achieve a specific outcome under specified circumstances over a certain time interval (this in contrast to simple static collection of screens and menus: The scenario explicitly includes a time dimension of what happens when)[Ref. 23]". By performing the tasks, the test users should find usability problems and document them.

*Wide perception***:** In this method, the test user articulates his/her problem, impressions and questions which arise during the usability test. The test user is guided by an observer who quotes these statements and analyzes it afterwards.

*Taking snapshots***:** The analysis of both methods is quite time taking. So, they are only recommended if there are not many test users. Hence the facts are to be recorded frequently.

*Questionnaire***:** By filling out a questionnaire, the test user can communicate his experiences from testing the website.

There are other usability testing methods, and most of them can be combined to improve the evaluation. A good approach would be: First select representative test users and set up time and location for the usability test. Then introduce the test and give the users the tasks which they have to perform. Invite them to attend the tasks and collect data via videotape, notes or data logging. Conclude the test with a questionnaire and compensate the test users for the participation afterwards [Ref. 23].

VIII. Conclusion

The Internet and its applications become an integral part of our world and the large number of web users increases day by day. It supports the collaboration of distributed project teams and web-based project management tools are a valuable complement to project management. But web-based distributed applications have to be implemented as well in this distributive environment. This paper attempted to analyze the different assessment criterion for the development of web-based software projects. However, efforts can also make to introduce different sophisticated frameworks to support Distributed software development free from all complexities.

References

[1] Apache: The Apache DB project: ObJectRelationalBridge OJB. http://db.apache.org/ojb.
[2] Apache: Struts. http://struts.apache.org.
[3] Apache: Tiles guide. http://struts.apache.org/userGuide/dev_tiles.html.
[4] Apache: What is Torque? http://db.apache.org/torque.
[5] Alexander Schatten, Marian Schedenig, Amin And jomshoa: Open Science Workplace: Building an Web-Based Open Source Tool to Enhance Project Management, Monitoring and Collaboration in Scientific Projects.
[6] Eclipse.org. http://www.eclipse.org/.
[7] Galileo computing: Java ist auch eine Insel. http://www.galileocomputing.de/openbook/javainsel4.
[8] James Holmes: Struts console. http://www.jamesholmes.com/struts/console.
[9] Iana: Internet assigned numbers authority. http://www.iana.org. Ibim: Warum Project management.
[10] IETF: Post Office Protocol Version 3. http://www.ietf.org/rfc/rfc1939.txt.
[11] IETF: Simple Mail Transfer Protocol. http://www.ietf.org/rfc/rfc821.txt.
[12] IETF: Standard for the format of ARPA internet text messages. http://www.ietf.org/rfc/rfc822.txt.
[13] Josef Altmann, Gustav Pomberger: Cooperative Software Development: Concepts, Model and Tools / C. Doppler Laboratory for Software Engineering. Johannes Kepler University Linz, 2009.
[14] JBoss: JBoss application server. http://www.jboss.org.
[15] LEO: LEO English-German Dictionary. http://dict.leo.org.
[16] Sharman Lichtenstein: Knowledge development and creation in email. In: IEEE Proceedings of the 37th Annual Hawaii International Conference on System Sciences (2004), p. 245–254).
[17] LyX: LyX - The Document Processor. http://www.lyx.org/.
[18] Marco C. Bettoni, Robert Ottiger, Rolf Todesco, Kurt Zwimpfer: Individual knowledge management with MailTack. In: IEEE Proceedings of 13th International Workshop on Database and Expert Systems Applications (Sep. 2-6. 2002), p. 157–161. – Aix-en-Provence, France.
[19] J. Meads: Laid-off usability engineer, or why we don't get no respect. In: ACM Press. Interactions archive, Volume 9, Issue 3 (2002), p.13-16
[20] Martina Manhartsberger, Sabine Musil: Web Usability. Galileo Press GmbH, 2002.
[21] Nicholas C. Romano Jr., Fang Chen, Jay F. Nunamaker Jr.: Collaborative project management software. In: IEEE HICSS. Proceedings of the 35th Annual Hawaii International Conference on System Sciences (2002), p. 233 – 242.
[22] Jakob Nielsen: Usability 101: Introduction to Usability. http:// www.useit.com/alertbox/20030825.html.
[23] Jakob Nielsen: Usability 101: Severity Ratings for Usability Problems. http://www.useit.com/papers/heuristic/ severityrating.html.