Available online at http://www.journalijdr.com

IJDR

International Journal of
DEVELOPMENT RESEARCH

ISSN: 2230-9926

International Journal of Development Research
Vol. 06, Issue, 09, pp.9352-9360, September, 2016

Full Length Research Article

DIFFERENCES BETWWEN SOFTWARE AND WEB BASED SYSTEMS DEVELOPMENT

***[1]Ahmed Hassan Mohamed Ali and [2]Mohammad Nazir Ahmad**

[1]Faculty of Computer Science and Information Technology, University of Medical Sciences and Technology, Khartoum, Sudan
[2]Faculty of Computing, Universiti Teknologi Malaysia, Malaysia

ARTICLE INFO

Article History:
Received 12th June, 2016
Received in revised form
20th July, 2016
Accepted 27th August, 2016
Published online 30th September, 2016

Key Words:
Web based systems,
Software engineering models,
and Web engineering models.

ABSTRACT

It has been found that the conventional software engineering models such as for example Waterfall model, Prototyping, Incremental, Spiral, Rational Unified process (RUP) and Extreme programming (XP) cannot be used directly or not applicable for the development of web based systems (Kumar and Sangwan 2011). This is because Web applications have unique characteristics that make Web development different and more difficult than traditional software development. In this paper, the reasons of why software development methodologies failed in developing web based systems will be shown.

INTRODUCTION

Software is a computer program with associated documentation and configuration data which is needed to make these programs operate correctly (Sommerville 2004). There are many types of software but the major two types are:

- System software such as operating system.
- Application software helps perform some useful tasks such as games, information systems, real-time systems, embedded systems, office software and scientific software (James F. Peters and Pedrycz 2000; Kappel 2006).

Web based systems are those application software that deliver to users through Internet, intranets and extranets. The Internet is a worldwide collection of interconnected networks. An intranet is a private network inside a company using web-based applications, but for use only within an organization.An extranet is a private network that allows external access to customers and suppliers using web-based applications (Aranda 2007; Ingle and Meshram 2012).

Corresponding author: Ahmed Hassan Mohamed Ali
Faculty of Computer Science and Information Technology, University of Medical Sciences and Technology, Khartoum, Sudan

Web based systems currently is essential for business operation, marketing, and strategy (Lam 2011). Enterprises, travel and hospitality industries, banks, educational and training institutions, entertainment business and governments use large-scale web based systems and applications to improve, enhance and/or extend their operations. E-commerce has become global and widespread. Traditional legacy information and database systems are being progressively migrated to the Web. Modern Web based systems run on distributed hardware and heterogeneous computer systems. Furthermore, fuelled by recent advances in wireless technologies and portable computing and communication devices, a new wave of mobile Web applications are rapidly emerging. The Web has changed our lives and work at every level, and this trend will continue for the foreseeable future (Brandon and Dan 2008). There is no hesitation that the majority of information systems to be developed in the future will be web based even for internal purposes and that is coming from two reasons (Ziemer 2007; 2009):

- **Web based systems are more accessible.** The HTTP protocol used in web based systems is a standard protocol that can travel across corporate firewalls. The only client software a user need is a web browser. Also, web based systems are available on many platforms.

Web browsers are packaged with most operating systems these days.

- **Web based systems have a lower maintenance and deployment costs.** Since Web based systems are running in web browser, they do not depend on installing client software on each user's computer. Web based systems can be maintained by modifying code that resides on a server. This reduces the time and the cost of upgrade and deployment of web based systems compared to traditional client/server applications.

Categories of Web based Systems

Web based systems can be categorized in many ways — there is no unique or widely accepted way. Categorization of web based systems based on functionality (Table 2.1) is useful in understanding their requirements and for developing and deploying Web-based systems and applications (Suh 2005).

Table 1. Categories of web based systems based on functionality. Adapted from (Suh 2005)

Functionality/Category	Examples
Informational	Online newspapers, product catalogues, newsletters, manuals, reports, online classifieds, online books.
Interactive	Registration forms, customized information, Presentation, online games.
Transactional	Online shopping (ordering goods and services), online banking, online airline reservation, online payment of bills.
Workflow oriented	Online planning and scheduling, inventory Management, status monitoring, supply chain management.
Collaborative work environments	Distributed authoring systems, collaborative design tools
Online communities, marketplaces	Discussion groups, recommender systems, online marketplaces, e-malls (electronic shopping malls), online auctions, intermediaries.

Also (Ziemer 2004) shows the characteristics of simple and advanced web based systems in Table 2.2.

Table 2. Characteristics of simple and advanced web based systems

Simple Web Based Systems	Advanced Web Based Systems
Simple Web pages primarily presenting textual information	Complex Web pages
Information content doesn't change–fairly static	Information is dynamic–changes with time and users' need
Simple navigation	Difficult to navigate and find information
Stand-alone systems	Integrated with database and other planning, scheduling, and tracking systems
High performance isn't a major requirement	Required high performance and continuous availability
Developed by a single individual or by a small team	Requires a large development team with expertise in diverse areas
Used for information dissemination innoncore application	Deployed in mission-critical applications

Software development and Web based Systems Development

It has been found that the conventional software engineering models such as for example Waterfall model, Prototyping, Incremental, Spiral, Rational Unified process (RUP) and Extreme programming (XP) cannot be used directly or not applicable for the development of web based systems (Kumar and Sangwan 2011) . This is because Web applications have unique characteristics that make Web development different and more difficult than traditional software development, Also many of the web based system development activities are business oriented (Eldai 2008) for example web application are sales-oriented, web application and intranets are content-oriented. (Mendes and Mosley 2006) grouped the differences between Web and software development into 12 areas, which are as follows:

Application Characteristics: Web applications are created by integrating several different elements, such as fine-grained components (e.g. DCOM, OLE, ActiveX), interpreted scripting languages, components off the shelf (COTS) (e.g. customized applications, library components, third-party products), multimedia files (e.g. audio, video, 3D objects), HTML/SGML/XML files, graphical images, mixtures of HTML and programs, and databases (Reifer 2000; Deshpande and Hansen 2001; Offutt 2002). In contrast, conventional software applications can also be developed using a wide variety of components (e.g. COTS), generally developed using conventional programming languages such as C++, Visual Basic, and Delphi.

Primary Technologies Used: Web applications are developed using a diverse technology such as the many flavored Java solutions, HTML, JavaScript, XML, UML, databases, and much more. In contrast, the primary technology used to develop conventional software applications is mostly represented by object-oriented methods, generators, and languages, relational databases, and CASE tools (Reifer 2000).

Approach to Quality Delivered: For Web development, quality is often considered as higher priority than time to market, with the mantra "later and better" as the mission statement for Web companies which wish to remain competitive (Offutt 2002). Within the context of conventional software development, software contractors are often paid for their delivered application regardless of its quality. They are also often paid for fixing defects in the delivered application, where these failures principally exist because the developer did not test the application thoroughly (Mendes and Mosley 2006).

Development Process Drivers: The dominant development process drivers for Web companies are composed of seven quality criteria (Offutt 2002): Reliability, Usability, Security, Availability, Scalability, Maintainability and Time to Market. With regards to conventional software development, the development process driver is time to market not quality criteria (Offutt 2002).

Availability of the Application: Customers who use the Web expect applications to be operational throughout the whole year (24/7/365). Any downtime, no matter how short, can be detrimental (Offutt 2002). Customers of conventional software applications do not expect these applications to be available 24/7/365.

Customers (Stakeholders): Web applications can be developed for use of a single organization (intranet), a number of organizations (extranets), or for use by people anywhere in the world. The implications are that stakeholders may come from a wide range of groups where some may be clearly identified and some may remain unknown, which is often the case (Mendes and Mosley 2003). As a result, Web developers are regularly facing the challenge of developing applications for unknown users, whose requirements and behavior patterns are also unknown at development time. With regards to conventional software applications, it is usual for stakeholders to be explicitly identified prior to development.

Update Rate (Maintenance Cycles): Web applications are updated frequently without specific releases and with maintenance cycles of days or even hours. In addition, their content and functionality may also change significantly from one moment to another, and so the concept of project completion may seem unsuitable in such circumstances (Mendes and Mosley 2006). The maintenance cycle for conventional software applications complies with a more precise process (Hughes 2007).

People Involved in Development: Web applications usually require a team of people with diverse skills and experience (Lang 2009). Such teams consist of Web designers and programmers, graphic designers, librarians, database designers, project managers, network security experts, and usability experts. The development of conventional software requires IT professionals where knowledge of programming, database design, and project management is necessary.

Architecture and Network: Web applications are typically developed using a simple client–server architecture (two-tier), represented by Web browsers on client computers connecting to a Web server hosting the Web application, to more sophisticated configurations such as three-tier or even n-tier architecture (Maxwell 2002). Conventional software applications either run in isolation on a client machine or use two-tier architecture whenever applications use data from database systems installed on a separate server (Mendes and Mosley 2006). The type of networks used by the stakeholders is usually known in advance since most conventional software applications are limited to specific places and organizations.

Disciplines Involved: To develop large and complex Web applications adequately, a team of people with a wide range of skills and expertise in different areas is required. These areas reflect distinct disciplines such as software engineering (development methodologies, project management, tools), hypermedia engineering (linking, navigation), requirements engineering, usability engineering, information engineering, graphics design, and network management (performance measurement and tuning) (Conte 2007; Gray 2007; Hughes 2007). Building a conventional software application involves contributions from a smaller number of disciplines than those used for developing Web applications, such as software engineering, requirements engineering, and usability engineering.

Legal, Social, and Ethical Issues: The Web as a spread environment enables a huge amount of structured (e.g. database records) and unstructured (e.g. text, images, audio) content to be easily available to a multitude of users worldwide. This is often cited as one of the greatest advantages of using the Web. However, this environment is also used for the purpose of dishonest actions, such as copying content from Web applications without acknowledging the source, distributing information about customers without their consent, infringing copyright and intellectual property rights, and even, in some instances, identity theft (Mendes and Mosley 2006). Conventional software applications also share a similar chance to that of Web applications, although to a smaller extent, these applications are not so readily available for such a large community of users, compared to Web applications.

Information Structuring and Design: As previously mentioned, Web applications present structured and unstructured content, which may be distributed over multiple sites and use different systems (e.g. database systems, file systems, multimedia storage devices) (Finnie 2007). In addition, the design of a Web application, unlike that of conventional software applications, includes the organization of content into navigational structures by means of hyperlinks. These structures provide users with easily navigable Web applications. Well-designed applications should allow for suitable navigation structures (Conte 2007), as well as the structuring of content, which should take into account its efficient and reliable management. Another difference between Web and conventional applications is that Web applications often contain a variety of specific file formats for multimedia content (e.g. graphics, sound, and animation). These files must be integrated into any current configuration management system, and their maintenance routine also needs to be organized as it is likely that it will differ from the maintenance routine used for text-based documents (Briand 2005). Conventional software applications present structured content that uses file or database systems. The structuring of such content has been addressed by software engineering in the past so the methods employed here for information structuring and design are well known by IT professionals (Mendes and Mosley 2006).

Challenges of Web based Systems Development

(Suh 2005) addresses the challenges that are facing developers in web based system development, regardless of its types, as follows:

- **Usability design:** The usability of an e-commerce Web site, to a large extent, will determine the success or failure of the organization's Web presence.
- **Content rich:** content rich web based systems requires frequent update and maintenance.
- **Scalability:** An Internet application runs in diverse operating environment than a non-Internet based application does.
- **Load Balancing:** In a multi-server Internet application, unbalanced workload on servers reduces system performance, reliability, and availability. Balancing system's load requires careful selection from an array of tools and techniques.

- **Security:** Security is a major concern for Internet applications because of the open operating environment.
- **Integrating Legacy Systems:** many organizations are linking their legacy systems, which may run on different computing platforms, to their Web applications.
- **Fast Development:** web developer should produce high quality web based systems in a very short period of time.

Software Process Models and Web based System Development

Software process models can be classified into two groups based on its focus:

- **Traditional software process models** focus on process over people and put more stress on documentation such as waterfall, prototyping, incremental, spiral, rational unified process (RUP).
- **Agile software process models** focus on people over process and put more stress on delivering functionality at end of the each iteration such as extreme programming (XP).

Waterfall

The Waterfall Model was first Process Model was used in software engineering to ensure success of the software development. Waterfall model contains six phases, each phase must be completed before the next phase can begin and there is no overlapping in the phases and each phase output is considered as input for the next phase. That's why is called linear sequential model.

The waterfall model consists of six phases:

- Requirements analysis phase: in this phase all system to be developed, requirements are captured and documented in requirements specification document.
- System and software design phase: in this phase system design is prepared which helps in specifying hardware, system requirements and identifying overall system architecture.
- Implementation and unit testing phase: in this phase system is developed in small units and each unit is tested for its functionality.
- Integration and system testing phase: all units were developed in the previous phase is integrated into system and then the entire system is tested.
- Deployment phase: in this phase system is deployed in the customer environment.
- Maintenance phase: all issues come up in the client environment is fixed in this phase

Suitability of waterfall model for web based systems development

Based on waterfall model and recommended practices for web development, we can conclude that (Winston 1987) (Howcroft and Carroll 2007) (Munassar and Govardhan 2010) (Silberberg 2006):

- Waterfall model cannot respond to rapid changing in web based systems requirements because it's not flexible.
- Problems are often not discovered until system testing and that makes maintenance cost is high.
- Cannot handle short development life cycle or parallel development of different release.

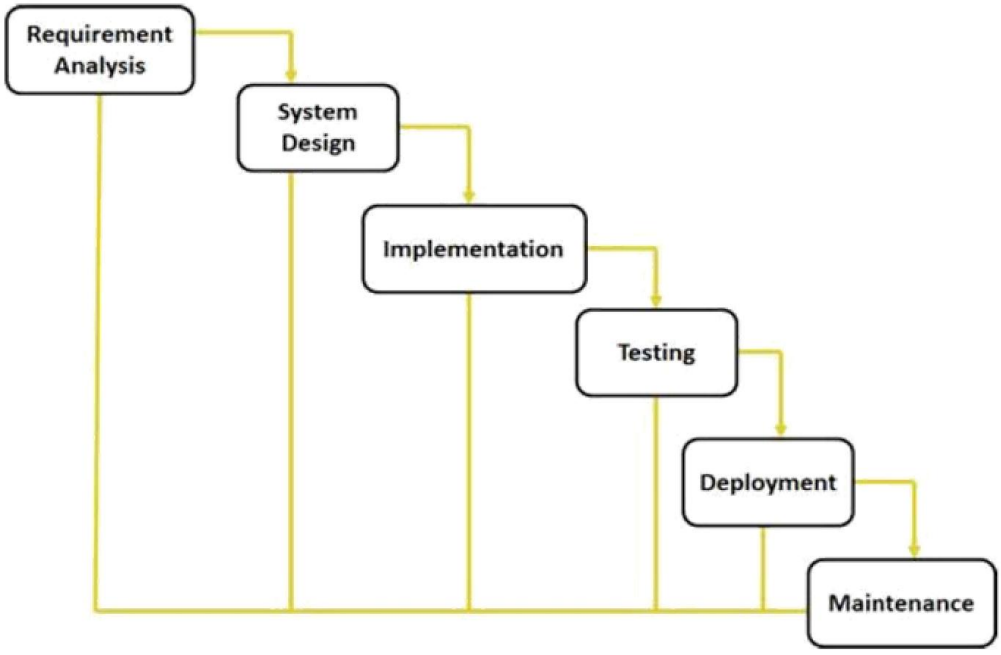

Figure 1. Waterfall model

- We can benefit from the idea of phases to organize activities and breaking down complex task into smaller tasks.

Prototyping

Prototyping is a development methodology in which a model is quickly constructed to test or illustrate design features and ideas, in order to gather user feedback. The main idea in this model is to build throwaway prototype relying on current known requirements then released to the user to understand his/her requirements. Prototyping is an attractive idea for complicated and large systems for which there is no manual process or existing system to help determining the requirements. The prototype is usually not complete systems and many of the details are not built in the prototype. This model consists of two types:

- **Throwaway prototype:** result system specification.
- **Evolutionary Prototype:** result final system.

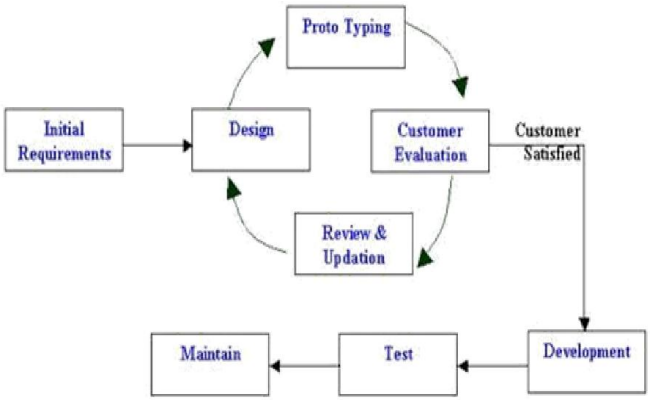

Figure 2. Prototype

Suitability of prototyping model for web based systems development

Based on nature of web based systems and recommended practices for web systems development we can conclude that (Bell 2000; Howcroft and Carroll 2007; Centers for Medicare & Medicaid Services 2008) prototyping model can handle changing requirements, reduce time and cost of maintenance. On the other hand, prototyping model cannot handle short time development cycle, cannot handle parallel development of different releases and may increase the complexity of the system as scope of the system may expand beyond original plans.

Incremental

In incremental model the whole requirement is divided into various builds. Multiple development cycles take place here, making the life cycle a "multi-waterfall" cycle. Cycles are divided up into smaller, more easily managed modules. Each module passes through the requirements, design, implementation and testing phases. A working version of software is produced during the first module, so you have working software early on during the software life cycle. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved. The model main principles are (Centers for Medicare & Medicaid Services 2008):

- A series of mini-Waterfalls are performed, where all phases of the Waterfall development model are completed for a small part of the system, before proceeding to the next increment.
- Overall requirements are defined before proceeding to evolutionary, mini waterfall development of individual of the system.
- The initial software concept, requirements analysis, and design of architecture and system core are defined using the Waterfall approach, followed by iterative prototyping, which culminates in installation of the final prototyping.

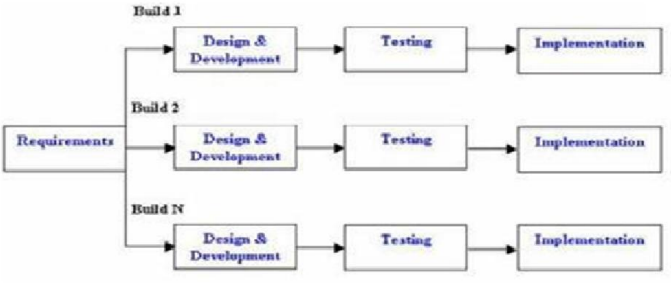

Figure 3. Incremental model

Suitability of incremental model for web based systems development

Based on nature of web based systems and recommended practices for web systems development we can conclude that (Centers for Medicare & Medicaid Services 2008; Munassar and Govardhan 2010) incremental model needs good planning and design, needs a clear and complete definition of the whole system before it can be broken down and built incrementally. There is usually lack of overall consideration of the business problem and technical requirements for the overall system and it cannot respond to rapid changing in requirements.

Spiral

This model combines ideas from other models through geting the benefits from iterative nature of Prototype with the controlled aspects of Waterfall model to provide rapid development of incremental versions of the software. In each iteration of the spiral approach, software development process follows the phase-wise linear approach. At the end of first iteration, the customer evaluates the software and provides the feedback. Based on the feedback, software development process enters into the next iteration and subsequently follows the linear approach to implement the feedback suggested by the customer. The process of iteration continues throughout the life of the software. Spiral Approach Phases:

- Customer Communication: Includes understanding the system requirements by continuous communication between the customer and the system analyst.
- Planning: Includes estimating schedule, cost, and resource for the iteration.

- Risk Analysis: includes identifying, estimating, and monitoring technical and management risks, such as schedule slippage and cost overrun.
- Engineering: Includes requirement gathering and design of the software system.
- Construction and release: Includes coding, testing and deploying software at the customer site and providing user-support documents.
- Customer Evaluation: Includes evaluation of software by the customer and implementing feedback in the next iteration of the software development.

Suitability of spiral model for web based systems development

Based on nature of web based systems and recommended practices for web systems development we can conclude that spiral model (Silberberg 2006; Munassar and Govardhan 2010) need skilled and experienced project manager, can be a costly model to use, doesn't work well for smaller projects, risk analysis requires highly specific expertise and lack of milestones.

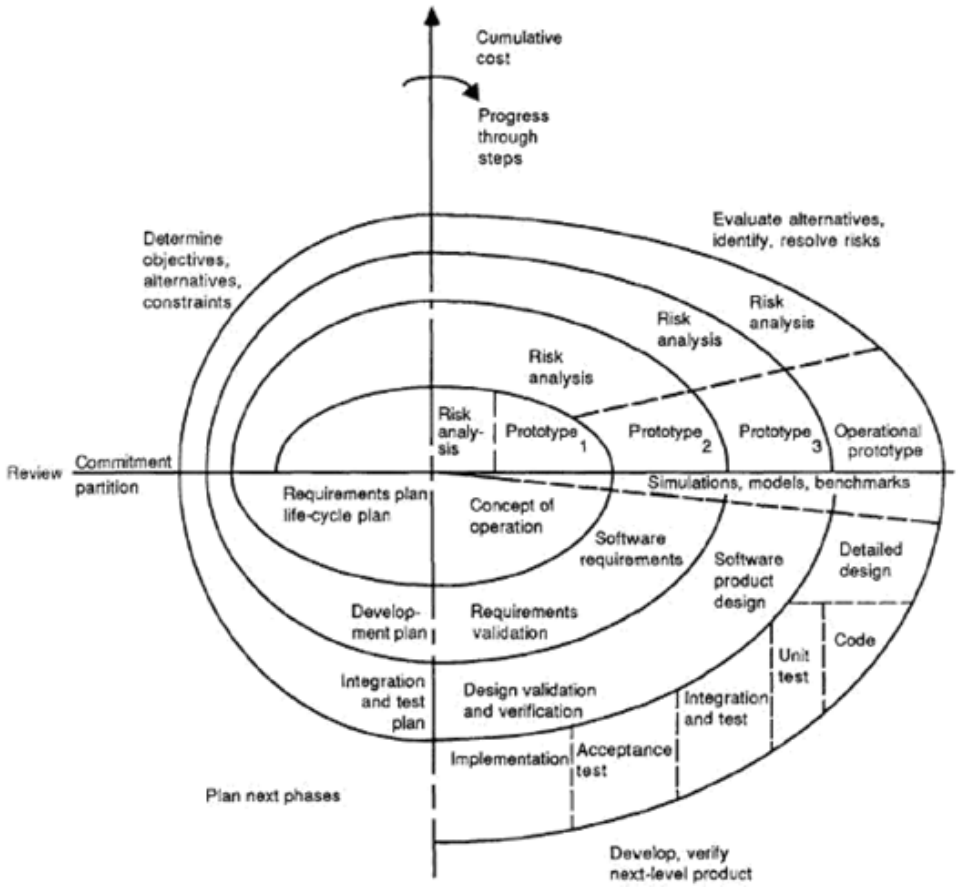

Figure 4. Spiral Model

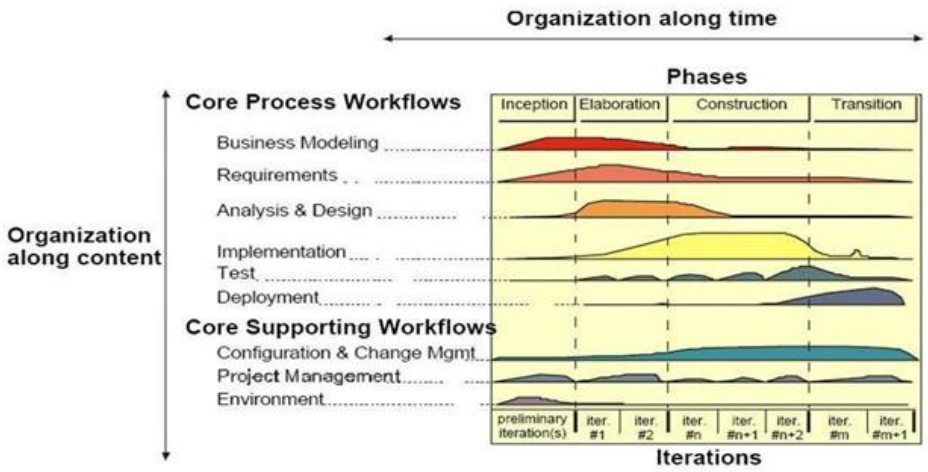

Figure 5. RUP (Sven Casteleyn, Florian Daniel *et al*. 2009)

Rational Unified Process Model (RUP)

The RUP is an iterative software development process created by the Rational Software Corporation in the 1980s and 1990s. According to the RUP, software is built along several cycles; figure 2.8 shows high level view of the process. Each cycle when finished produces release and is executed in four separate phases (Sven Casteleyn, Florian Daniel et al. 2009):

- **Inception:** in this phase, general system idea is established and critical use cases are developed.
- **Elaboration:** in this phase, most critical cases are released and project manager should be able to justify the resources needed for the project, budget and human resources.
- **Construction:** in this phase, complete product is developed and all needed use cases are realized.
- **Transition:** in this phase, small group of experienced users test the product and suggests improvements and figure out the defects. Also this phase involves users training.

Suitability of RUP model for web based systems development

Based on nature of web based systems and recommended practices for web systems development we can conclude that this model (Silberberg 2006; Centers for Medicare & Medicaid Services 2008; Munassar and Govardhan 2010) need skilled and experienced project manager, can be a costly model to use and doesn't work well for smaller projects.

The Rapid Application Development (RAD) Model

The RAD is an incremental software process model that emphasizes a short development cycle. In RAD model the components or functions are developed in parallel as if they were mini projects. The developments are time boxed, delivered and then assembled into a working prototype. This model is a high-speed adaptation of waterfall model in which rapid development is achieved by using a component based construction approach.

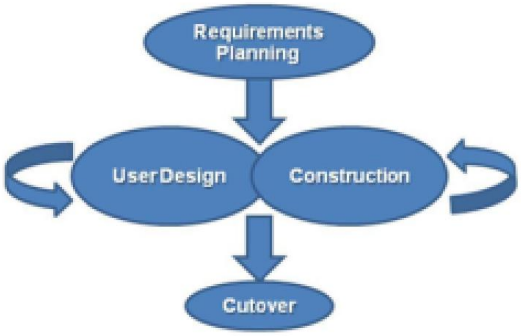

Figure 6. RAD Model

The phases in the rapid application development (RAD) model are (Hough 1993):

- **Business modeling:** The information flow is identified between various business functions.
- **Data modeling:** Information gathered from business modeling is used to define data objects that are needed for the business.
- **Process modeling:** Data objects defined in data modeling are converted to achieve the business information flow to achieve some specific business objective.
- **Application generation:** Automated tools are used to convert process models into code and the actual system.
- **Testing and turnover:** Test new components and all the interfaces.

Suitability of RAD model for web based systems development

It depends on strong team and individual performances for identifying business requirements. The only system that can be modularized can be built using RAD, requires highly skilled developers/designers, high dependency on modeling skills and Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.

Extreme Programming (XP)

XP model is focusing on delivering small increments of functionality. It relies on constant code enhancements, user involvement in development team and pair programming.

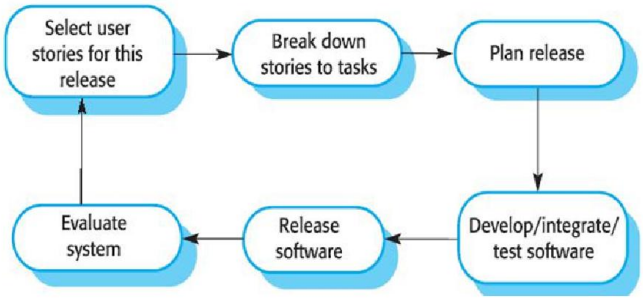

Figure 7. XP Model (Munassar and Govardhan 2010)

XP and Agile principles

- Small, frequent system releases supported through incremental development.
- Customer involvement.
- Pair programming, collective ownership and a process that avoids long working hours.
- Change supported through regular system releases.
- Maintaining simplicity through constant refactoring of code.

Suitability of XP model for web based systems development

Based on nature of web based systems and recommended practices for web systems development we can conclude that XP concepts need some adaption to support web engineering frame work processes (Ali 2008), needs experience and skill if not to degenerate into code-and-fix (Munassar and Govardhan 2010), difficult to scale up to large projects (Munassar and Govardhan 2010) and programming pairs is costly.

DISCUSSION

Most researchers agree that web based systems are different from most other types of software systems because of the web based systems unique characteristics. The unique characteristics of web based systems are (Pressman and Lowe 2009):

- **Network intensiveness**: It resides on a network and must serve the needs of a diverse community of clients.
- **Concurrency**: large number of users can access the Web application at the same time.
- **Unpredictable load**: the number of users of the Web application may vary by orders of size from day to day.
- **Performance**: users of Web application should not wait long time for Web application response.
- **Availability**: users of Web application often demand access on "24/7/365" basis.
- **Data driven**: In many cases, the primary function of a Web application is to use hypermedia to present text, graphics, audio, and video content to the end user.

Table 3. Software Engineering models weaknesses

Model	Weaknesses
Waterfall	• Cannot respond to rapid changing in requirements (Howcroft and Carroll 2007). • Inflexible (Munassar and Govardhan 2010). • System performance cannot be tested until the system is nearly fully coded (Munassar and Govardhan 2010). • Problems are often not discovered until system testing (Munassar and Govardhan 2010). • Difficult and expensive to make changes (Silberberg 2006).
Prototyping	• Requirements may frequently change significantly (Centers for Medicare & Medicaid Services 2008). • Can lead to poorly designed systems (Howcroft and Carroll 2007). • Identification of non-functional requirement is difficult to document (Centers for Medicare & Medicaid Services 2008)
Incremental (Centers for Medicare & Medicaid Services 2008)	• There is usually lack of overall consideration of the business problem and technical requirements for the overall system. • Cannot respond to rapid changing in requirements. • Did not support all web engineering frame work processes.
Spiral	• Need skilled and experienced project manager (Centers for Medicare & Medicaid Services 2008). • Can be a costly model to use (Silberberg 2006). • Doesn't work well for smaller projects (Munassar and Govardhan 2010).
XP	• XP concepts need some adaption to support web engineering frame work processes (Ali 2008). • Needs experience and skill if not to degenerate into codeand-fix (Munassar and Govardhan 2010). • Difficult to scale up to large projects (Munassar and Govardhan 2010)
RUP (Kappel 2006)	• Did not support short development cycles. • Cannot respond to rapid changing in requirements.

Content sensitive: the quality and aesthetic nature of content remains an important determinant of the quality of Web application.

- **Continuous evolution**: It is not unusual for some Web application (specifically, their content) to be updated on an hourly schedule.
- **Immediacy**: the time to market for a complete Web site can be a matter of a few days or weeks.
- **Security**: In order to protect sensitive content and provide secure modes of data transmission strong security actions must be implemented throughout the infrastructure that supports a Web application and within the application itself.
- **Aesthetics**: (Lavie and Tractinsky 2004) called the sophisticated use of web technology "web-based aesthetics". For example, researchers have identified web based aesthetics as often including advanced graphics and multimedia features such as sound, animation and video streaming (Bansler 2000; Kautz, 2007).

Conclusion

The differences listed in the previous section; result in some Development practices that are special for web based systems, which are (Ahmed Ali and Ahmad 2015):

Parallel development: To meet the short Time-to-Market requirement, developers have to try to shorten the time needed to develop a new release. This can only be done by parallel development, meaning that the development team is working on two different releases simultaneously.

Release orientation: In the early phases of a Web Application, releases are made in short cycles. The release cycle can be between 2 and 15 days (Ramesh 2002). A new release is defined by its deadline, and not by its content. It is more important to keep the deadline than to fulfill the expected requirements for a new release.

Tool dependence: Many Web Application development organizations make heavy use of development tools to speed up the design and coding process (Ziemer 2004).

Customer involvement: Since the requirements are evolving during the development, customers are involved intimately in the development effort.

Prototyping: Prototypes are used to deal with the unstable and evolving requirements. They are used to agree on requirements, and to receive feedback both from the customer and from the end users.

Increment and iterative concepts should be used in development because of requirements evolving.

Author contribution

Developing web based systems is different from developing conventional software due to many reasons, those reasons were shown in this paper.

Acknowledgment

This paper research is a part of my PhD thesis work on web engineering and support from university of medical sciences and technology is gratefully acknowledged.

REFERENCES

"This book presents current, effective software engineering methods for the design and development of modern Web-based applications"--Provided by publisher.

Ahmed Ali and M. Ahmad, 2015. "Recommended practises for developing web based systems." *International Journal of Development Research* 1.

Ali, A. H. M. H. 2008. "A novel Methodology for Developing Web Based Systems". International Conference on Software Engineering Theory and Practice (SETP08). Orlando, FL, USA.

Aranda, P. J. V. 2007. Requirements Engineering Approach for the Development of Web Applications. Department of Information Systems and Computation, Technical University of Valencia. PhD: 360

Bansler, J. 2000. "Corporate Intranet Implementation: Managing emergent technologies and organizational practices." AIS 1: 1-39.

Bell, D. 2000. Software Engineering: A Programming Approach, Addison Wesley.

Brandon and Dan, 2008. "Software engineering for modern Web applications : methodologies and technologies". Hershey, PA, Information Science Reference.

Briand, L. 2005. "An Assessment and Comparison of Common Cost Estimation Modeling Techniques". ICSE, Los Angeles, USA.

Centers for Medicare & Medicaid Services, D. o. H. a. H. S. 2008. "Selecting a development approach." 10.

Conte, S. 2007. ""Software Engineering Metrics and Models"." Benjamin/Cummings.

Deshpande, Y. and S. Hansen 2001. "Web engineering: creating a discipline among disciplines." IEEE Multimedia 8(2): 8-87.

Eldai, O. I. 2008. "Towards a New Methodology for Developing Web-Based Systems." World Academy of Science, Engineering and Technology 46.

Finnie, G. 2007. "A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models." Systems and Software 39: 281-289.

Gray, R. 2007. "Factors Systematically associated with errors in subjective estimates of software development effort: the stability of expert judgement". the IEEE Metrics Symposium.

Hough, D. 1993. "Rapid Delivery: An evolutionary approach for application development." *IBM Systems Journal* 32(3): 397-419.

Howcroft, D. and J. Carroll, 2007. "Proposed Methodology for Web Development". *IS Research Centre*, University of Salford.

Hughes, R. 2007. "An Empirical investigation into the estimation of software development effort". Dept. of Computing, University of Brighton. PhD.

Ingle, D. and D. B. B. Meshram, 2012. "Hybrid Analysis and Design Model for Building Web Information System." *IJCSI International Journal of Computer Science Issues* 9(4).

James, F. Peters and W. Pedrycz. 2000 Software Engineering: An Engineering Approach. New York, John Wiley & Sons.

Kappel, G. 2006. Web engineering: the discipline of systematic development of web applications. Hoboken, NJ, John Wiley & Sons.

Kautz, K. 2007. "Persistent problems and practices in information systems development." Information Systems 17: 217–239.

Kumar, S. and S. Sangwan. 2011. " Adapting the Software Engineering Process to Web Engineering Process." *International Journal of Computing and Business Research* 2(1 2011).

Lam, M. 2011. "Methodologies, tools, and techniques in practice for web application development." *Journal of Technology Research* 3.

Lang, 2009. "Web-based Systems Development: The Influence of Disciplinary Backgrounds on Design Practices." JIOS 33(1).

Lavie, T. and N. Tractinsky, 2004. "Assesing dimensions of perceived visual aesthetics of web sites." *International Journal Human-Computer Studies* 60: 269-298.

Maxwell, K. 2002. "Applied Statistics for Software Managers." Englewood Cliffs.

Mendes and Mosley, 2003. "A Replicated Assessment of the Use of Adaptation Rules to Improve Web Cost Estimation". ACM and IEEE International Symposium on Empirical Software Engineering, Rome, Italy.

Mendes and Mosley, 2006. "Web engineering". Berlin ; New York, Springer.

Munassar, N. M. A. and A. Govardhan, 2010. "A Comparison Between Five Models Of Software Engineering." IJCSI *International Journal of Computer Science* 7(5).

Offutt, J. 2002. "Quality attributes of Web software applications." IEEE Software 19(2): 25-32.

Pressman, R. S. and D. B. Lowe, 2009. "Web engineering : a practitioner's approach". Boston, McGraw-Hill Higher Education.

Ramesh, B. 2002. "Internet software engineering: A different class of processes." *Annals of Software Engineering* 14(4): 169–195.

Reifer, D. 2000. "Web development: estimating quick-to-market software." IEEE Software: 57-64.

Silberberg, R. 2006. "An investigation into methods used to develop software systems." Elektron.

Sommerville, I. 2004. Software Engineering, Pearson Addison Wesley.

Suh, W. 2005. "Web engineering : principles and techniques". Hershey, PA, Idea Group Pub.

Sven Casteleyn, Florian Daniel, et al. 2009. Engineering Web Applications. New York, Springer.

Technical session SW: 2009. Software and web engineering [breaker page]. Networking and Media Convergence, 2009. ICNM 2009. *International Conference on*.

Winston, R. 1987. Managing the Development of Large Software Systems: Concepts and Techniques. International Conference on Software Engineering, California, United States, IEEE Computer Society Press Los Alamitos.

Ziemer, 2004. Web Engineering. Computer Science Graduate Student Conference 2004. Norway.

Ziemer, 2007. "An Architecture for Web Applications." Essay in DIF 8914 Distributed Information Systems.

****