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

**International Journal of Innovative Research in Computer and Communication Engineering**

*(An ISO 3297: 2007 Certified Organization)*

**Vol. 4, Issue 5, May 2016**

# An Analysis of Web-Based Application Development Models

Manpreet kaur

Asst. Professor, Dept. of Computer Science, S.R. Govt. College, Amritsar, Punjab, India

**ABSTRACT**: The web has evolved into a global environment for delivering all kinds of applications, ranging from small-scale and short-lived services to large-scale, enterprise workflow systems distributed over many servers [5]. Web-based applications need to be reliable and perform well. To build such applications, Web developers need a sound methodology, a disciplined and repeatable process, better development tools, and a set of good guidelines[2]. The paper identifies and analyzes various aspects of conventional and Web-based development. Different Web-based development methodologies based on traditional web development approach, object-oriented approach and hybrid model incorporating conventional software development approaches have been discussed. It has also been identified that the hybrid model is a better approach than individual Waterfall model or Spiral model for fulfilling the expectations of advanced Web Application development life cycle.

**KEYWORDS:** Web-based Application, Web Application development life cycle, Web Implementation Model, Web Composition Component Model, Agile-based approach, Plan-driven methodology.

## I. INTRODUCTION

Web-based applications are becoming so prevalent that every single day people would not go without using them. The applications range from simple to extensive, resulting in huge amount of revenue generation. Therefore in immense pleasure development of applications is becoming a challenging task. Although the development of Web-based applications made many improvements, there is still a lack of an established software engineering methodology for constructing Web-based applications [10]. There is no rigorous, systematic approach, and most current Web application development and management practices rely on the knowledge and experience of individual developers [5].

Web-based applications differ from other applications due to its high reliability, high usability, more secured, incorporate advanced technologies, takes a shorter time to market, have shorter product life cycles and require continuous maintenance [3]. A survey on Web-based projects, published by the Cutter Consortium in 2000, revealed a number of problems with large outsourced Web-based projects [4]:

- 84% of surveyed delivered projects did not meet business needs
- 53% of surveyed delivered projects did not provide the required functionality
- 79% of surveyed projects presented schedule delays
- 63% of surveyed projects exceeded their budget

Many software engineers have become concerned about the way Web-based applications are being developed [10]. In this paper, three models of developing web applications Web *Implementation model, Web Composition Component model* and a *hybrid model* incorporating Agile approach and Plan-driven approach are analyzed. Web Implementation model is used for modeling web delivery documents, whereas Web Composition model is based on object-oriented principles like abstraction, modularity, reuse, and Generality. This paper investigated that conventional software development methodologies can be incorporated together resulting into efficient web application development.

## II. INTRODUCTION TO WEB APPLICATIONS

Web-based applications are not just web pages. Several researchers have attempted to define Web-based applications. *Ceri et al.* defined Web-based applications as “complex systems, based on a variety of hardware and software components, protocols, languages, interfaces, and standards” [13]. *Conallen* defined Web-based applications as “a Web system (Web server, network, HTTP, browser) in which the user input (navigation and data input) affects the state of the business" [7]. *Gellersen* and *Gaedke* defined Web-based applications as any software application that depends on the Web for its correct execution [5]. *Koch* and *Kraus* defined Web-based applications as "Web information systems that tend to be used to integrate and streamline business processes across organizations (customers, agents, suppliers, others) and geographical borders" [1]. As the scope and complexity of current Web-based applications vary widely, it is difficult to come to a standard definition of Web-based applications. The characteristics of Web-based applications can be broadly considered into following issues as shown in Table 1 [10].

**Table 1: Classification of Web- based applications characteristics**

| **Issues** | **Characteristics** |
|---|---|
| ***Architecture*** | Network intensive, Content Driven, Distributed, Static and Dynamic, Hypermedia, Multiplatform Support |
| ***Design*** | Content driven, Autonomous |
| ***Performance*** | Concurrency, Availability, Time-to –market |
| ***Quality*** | Concurrency, Compatibility, Usability, Interoperability, Heterogeneity, Credibility |
| ***Security*** | Session Management, Credibility |
| ***Development*** | Continuous Evolution, Short Development, Schedule, Modularity, Maintainability |
| ***Aesthetics*** | Appealing Appearance, Graphic User Interface, Internationalization |

Based on their functionality Web-based applications can be grouped into various categories, and a given application could fall under more than one category [2].

**Table 2: Categories of Web Applications**

| ***Category*** | **Examples** |
|---|---|
| ***Informational*** | Online newspapers, product catalogs, newsletters, service manuals, online classifieds, online electronic books |
| ***Interactive*** | Registration forms, customized information, presentation, online games. |
| ***Transactional*** | Electronic shopping, ordering goods and services, online banking |
| ***Workflow*** | Online planning and scheduling systems, inventory management, status monitoring |
| ***Collaborative work Environments*** | Distributed authoring systems, collaborative design tools |
| ***Online Communities, Marketplaces*** | Chat groups, recommender systems that recommend products or services, online marketplaces, online auctions |
| ***Web portals*** | Electronic shopping malls, online intermediaries |

## 2.1 Web Application Life Cycle versus Software Development Life Cycle:

Web and software development differ in a number of areas. These areas encompass the people involved in the development of these applications, the intrinsic characteristics of the applications, and the audience for which they are

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

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

Vol. 4, Issue 5, May 2016

developed. Table 3 specifies the various differences between the Web-based application development approach and conventional development approach.

*Table 3: Web - based Approach vs. Conventional Approach to development [12]*

| | Web-Based Approach | Conventional Software Approach |
|---|---|---|
| **Application Characteristics and availability** | Integration of numerous distinct components(e.g., fine-grained, interpreted scripting languages ;COTS; multimedia files; HTML, SGML, and XML files; databases; graphical images) and distributed, cross-platform applications, and structuring of content using navigational structures with hyperlinks. Availability throughout the whole year. | Integration of distinct components ( e.g., COTS, databases, graphical images) and monolithic single-platform applications.<br><br>Except for a few application domains, no need for whole year availability. |
| **Technology and Architecture** | Variety of Java solutions (Java Servlets, Enterprise JavaBeans, applets, and JavaServer Pages), HTML, JavaScript, XML, UML, databases, third-party components and middleware, and so forth.<br>Architecture comprises two-tier to n-tier clients and servers with different network settings and bandwidth, sometimes unknown. | Object -oriented languages, relational databases, and CASE tools.<br><br>One-to two-tier architecture with network settings and bandwidth that are likely to be known in advance. |
| **Quality Drivers** | Quality is considered to be of higher priority than time to market.<br>Main quality drivers are reliability, usability, and security. | Time to market takes priority over quality. Main quality driver is time to market. |
| **Information Structuring, Design, and Maintenance** | Structured and unstructured content, use of hyperlinks to build navigational structures.<br>Maintenance cycles are frequent without specific releases.<br>Maintenance cycles of days or even hours. | Structured content with seldom use of hyperlinks.<br>Maintenance cycles are done via specific releases.<br>Maintenance cycles ranging from a week to several years. |
| **Disciplines and People involved in Development** | Disciplines are software engineering, hypermedia engineering, requirements engineering, usability engineering, information engineering, graphics design, and network management.<br>People are web designers and programmers, graphics designers, librarians, database designers, project managers, network security experts, usability experts, artists and writers. | Disciplines are software engineering, requirements engineering, and usability engineering.<br><br>People are IT professionals with knowledge of programming, database design, and project management. |
| **Stakeholders** | Wide range of groups, known, and unknown, residing locally or overseas. | Generally groups confined within the boundaries of departments, divisions, or organizations. |
| **Legal, Social, and Ethical issues** | Content can be easily copied and distributed without permission or acknowledgement of copyright and Intellectual property rights. Applications should take into account all groups of users including disabled people. | Content can also be copied infringing privacy, copyright, and IP (Internet Protocol) issues, albeit to a smaller extent. |

**2.2 Characteristics of Web Development Projects:**

Table 3 explains the basic characteristics of Web-based projects and how does they differ from Traditional projects.

*Table 3: Characteristics of Traditional versus Web development Projects [12]*

| **Characteristics** | **Traditional development** | **Web development** |
|---|---|---|
| **Primary objective** | Builds quality software products at minimum cost | Build quality products to market as quickly as possible |
| **Typical Project size** | Medium to large(hundreds of team members) | Small(3-5 team members) |
| **Typical timeline** | 12-18 months | 3-6 months |
| **Development approach Employed** | Classical, requirements-based, phased and/or incremental delivery, use cases, documentation-driven | Rapid application development, gluing building blocks together, prototyping, Rational Unified Process |
| **Primary engineering Technologies used** | Object-oriented methods, generators, modern programming languages (C++), CASE tools etc. | Component-based methods, 4th and 5th generation languages (html, java, etc.) visualization (motion, animation) etc |
| **Process employed** | CMM-Based | Ad hoc |
| **Products developed** | Code-based systems, mostly new, some reusable, many external interfaces, often complex. | Object-based systems, many reusable components (shopping carts, etc), few external interfaces, relatively simple |
| **People involved** | Professional software engineers with lots of experience | Graphic designers, less experienced software engineers |
| **Estimating technologies Used** | SLOC or function-point based models, WBS approach for small projects | Wing it |

## III. WEB APPLICATION DEVELOPMENT METHODOLIGIES

The paper discusses three different models for developing Web-based applications. These models have been explained as follows.

**3.1 Web Implementation Model:**

The World Wide Web was originally designed as an information medium for distributed research teams. A key objective was to make it as easy as possible for authors to deliver documents. For this kind of web application development, the life cycle comprises *informal analysis* of what is to be presented, *informal design* of how to structure it into hyperlinked chunks of information, and *implementation* through markup. Following implementation, the documents are *maintained* by the authors themselves. [5]

The ***Web implementation model*** was designed to meet these life-cycle requirements. It is deliberately simple, based on the notion of *resources* that model mostly self-contained chunks of information. Resources are authored and maintained rather independently of other resources, and links are the means by which resources can be combined into coherent sets of documents—for example, a Web site. The Web implementation model is based on flat decomposition of applications into resources. Resources have a unique address, and they are delivered on request from Web server to Web client. They can be static or dynamically generated from a script, but they are inherently specific, which means that they cannot capture abstractions.

The resource concept fits the document-development life cycle very well, and resources support the principles of modularity in this context. However, the use of the Web has moved far beyond its original scope, even in document delivery.

**Advancements in life cycle of Web Applications:**

- The life cycle is no longer author-centric.
- *Requirements analysis* involves, for example, information analysts as well as marketing people and other stakeholders.
- *Design* addresses both database development and graphic presentation.

- *Implementation* requires Web programmers to use features beyond simple markup.
- *Maintenance* involves site managers and Webmasters.

**Drawbacks of Web Implementation Model:**
Web implementation model does not meet the requirements of advanced life cycle of web applications. It is based on traditional software development approach suffers from various drawbacks:
- *Abstraction:* Web Implementation model does not support abstraction. The lack of abstraction means that a Web implementation cannot factor properties shared between objects into a generalized object but must build them into each specific object.
- *Reuse:* As Web Implementation model does not provide abstractions, so it does not capture structural design to reuse, even though layout and navigation structures are commonly reused in different parts of a site.
- *Modularity*: It does not support the modularity.
- *Redesign*: Another problem is the gap between higher level design and implementation. It is difficult to redesign during system evolution because a Web implementation cannot capture the concepts of the original design.
- *Irreversible*: The development process is not reversible, which means that design decisions are difficult to track and to access in the implementation.

In summary, the delivery of applications in the Web environment is radically different from the usual ways of delivering software, and imposes a completely different structure and approach on
application development. Current Web implementation technology is too low level to support a proper development process.

## 3.2. An Object-Oriented Model for Web Applications
WebComposition is an approach to structured Web development that applies established object-oriented software development principles to the World Wide Web. The approach is based on a Web component model that abstracts from low-level Web implementation technologies to support seamless, reversible development of Web applications. [5]
**WebComposition Component Model:**
WebComposition defines an object-oriented model that uses *components* as a uniform concept for modeling Web entities at arbitrary levels of granularity and abstraction. [5]. The arbitrary granularity means that an application can be decomposed to the actual units of change. In contrast to resources, components are not fixed to a certain grain size but designed instead to capture design artifacts at their natural granularity. Components can reference other components to model aggregation (has-part) or specialization (inherits-from). As another example, a component modeling a navigation structure can reference the components modeling the involved links and anchors.
By means of a special reference type, components can reference so-called *prototype components* from which they inherit state and behavior. Any component can function as a prototype. Thus, the WebComposition model is based on a *prototype instance paradigm*, which eliminates the distinction of instances and classes as known in most object models. This paradigm is naturally suited to Web application modeling, first because many Web entities—namely, those modeling content—are unique and, second, because prototyping reflects the copy-and-modify type of reuse often applied in Web development. Because it is easy to emulate a class-oriented view from prototypes, the Web-Composition model aligns conceptually with any object-oriented design model.

**Advantages of WebComposition Model:**

- The model is clearly defined, both on a conceptual level and as XML-based implementation technology in the WebComposition Markup Language (WCML).
- *Modifiable and Extensible*: New components can be added to WCML implementations of Web applications easily by reusing and modifying any component code in the system.
- *Abstraction and Reuse*: The WebComposition model supports abstraction, reuse modularity, and encapsulation.

## 3.3 Hybrid Model:
There have been numerous attempts to identify the important success factors and to use engineering approaches to building process models to address specific concerns [10].

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

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

Vol. 4, Issue 5, May 2016

**Incorporating Agile and Plan-driven approach:**
The lifecycle model contains various phases such as requirement processes, architecture envisioning, prototyping, feature driven development, Scrum processes, XP practices and repository management. In the model, some phases are performed iteratively to gain the desired outcome while the flow of the phases is sequential in nature [13].

**(i) Requirement Process**
The set of activities of the requirement process in Figure 1 include initial vision, information and requirements gathering and business analysis. At this stage, the project manager understands the vision of the customer. After understanding the initial vision of the customer, the project manager proceeds for requirement and information gathering. Once the requirements are collected, business analysis is performed. This phase considers how to compromise on functionality and performance to meet the deadline with the limited resources available in the organization decides the flow of phases for a project and produces a detailed plan for each phase.

**(ii) Architecture Envisioning**
Being a part of agile modeling, with architectural envisioning the team perform some high-level architectural modeling early in the project to help foster agreement regarding the technical strategy within the team and with

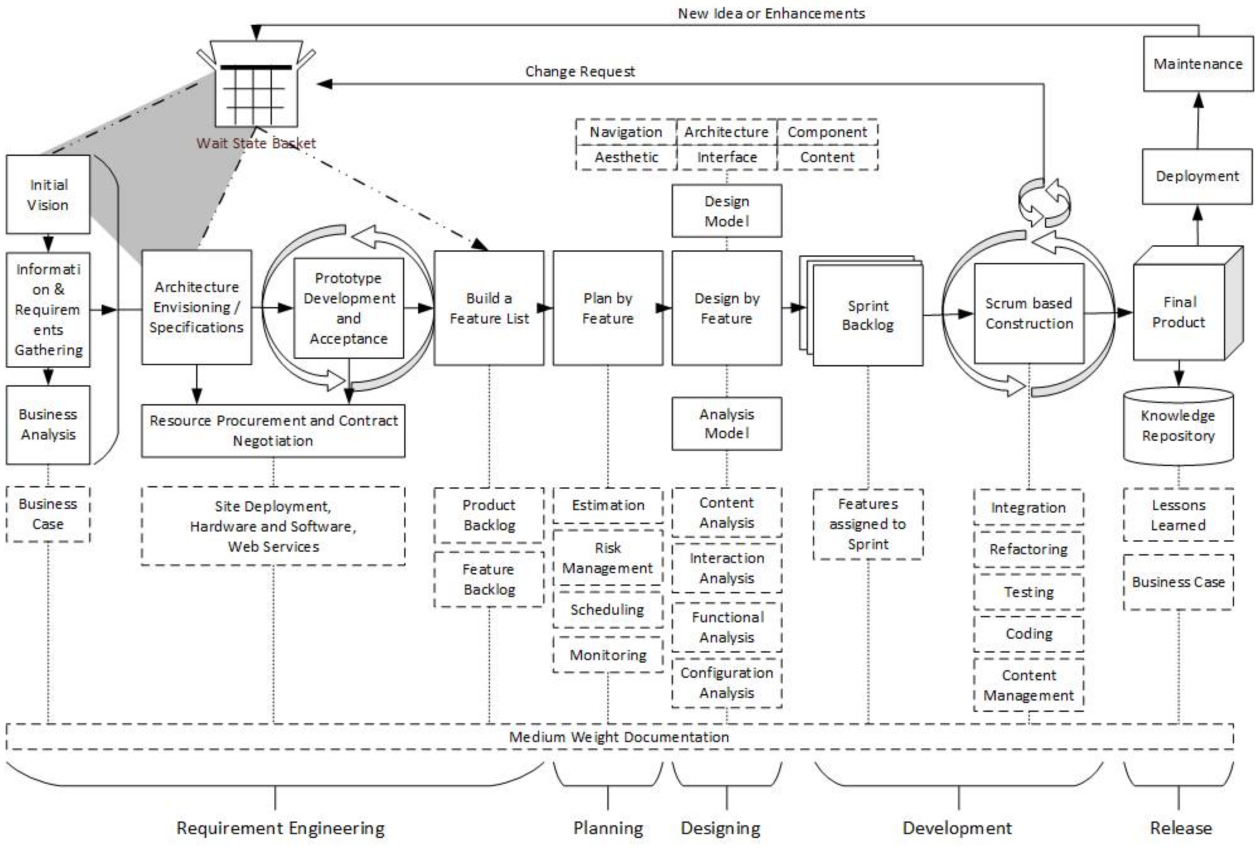

**Figure 1: A lifecycle model for Web-based application development**

critical stakeholders. The goal is to identify an architectural strategy and not to create a heavy documentation, enabling you to do this speedily. Architectural envisioning in an agile manner offers several benefits such as improved productivity, reduce technical risk, reduced development time, improved team communication and improved team organization

**(iii) Prototype Development and Acceptance**
Based on the user requirements and architecture envisioning, the team is able to create a prototype of the system. Focus is more on user interaction than low-level system functionality, such as database access. Benefits of creating a

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

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

Vol. 4, Issue 5, May 2016

prototype are, the software designer and developer can get valuable feedback from the users early in the project. The customer and stakeholders can visualize the prototype if it is able to meet the requirements. It also allows the software engineer some insight into the accuracy of initial project estimates and whether the deadlines and milestones proposed can be successfully met.

**(iv) Resource Procurement and Contract Negotiation**

The resources needed to deliver a product include people, machinery, materials, technology, property and anything else required to deliver the work. Resources may be obtained internally from the host organization or procured from external sources. The project manager must identify the resources required to deliver the work, as part of planning, and determine when the resources will be required, through scheduling. The acquisition of external resources will normally be through a procurement process that involves provider selection. This results in a contract for the provision of goods and services.

**(v) Feature Driven Development (FDD)**

The next three phase after prototype development and acceptance are derived from FDD. The short description of the phases derived from FDD are :

• *Build a Features List*: an initial project-wide activity to identify all the features to support the requirements.
• *Plan by Feature*: an initial project-wide activity to produce the development plan.
• *Design by Feature*: a per-feature activity to produce the feature design package.

**(vi) SCRUM**

After the design by feature phase, all the features of the Web-based applications to be developed in a sprint are collected as sprint backlog, from where the development process follows Scrum methodology. The sprint backlog, scrum based construction and product increment are derived from Scrum methodology. Scrum is a simple low overhead process for managing and tracking software development. The Scrum process begins with the creation of the Product Backlog comprising the prioritized product features required by the customer. The next phase of Scrum centre's upon a series of 30 day Scrum Sprints. During each Sprint, the Scrum team will complete a working set of features that have been selected (during a Scrum pre-Sprint planning session) from the overall Product Backlog. Short (e.g. 15 minute) meetings are held by the Scrum team on each day of the Scrum Sprint. Each daily meeting allows the team to monitor project status and discuss problems and issues. .

**(vii) Evolution Process**

Evolution processes include deployment, maintenance, and enhancements. In *deployment*, after being tested and evaluated, the Web-based application is deployed on the live server and then customers would be able to use the developed systems. *Maintenance* is to describe activities that address demands at code or near-code levels, which is applied for less complicated work due to the limited time and management control, i.e., to enable software to be fixed quickly so that the software can provide better services. If new ideas or *enhancements* arise during the development cycle, it is first sent to wait state basket. The concept of wait state basket is any changes approaching while development is first evaluated for its necessity. Unless the new idea or enhancements are felt to be very important for the initial launch of the web application, they reside in the basket and may be managed after the final deployment.

## IV. CONCLUSION

The phenomenal popularity of the Web and its advantages as a client-server software platform has led to a wide range of Web applications, but development is still largely ad hoc. *The Web implementation model* imposes a structure on Web applications that does not relate well to established software development models, rendering it difficult to adopt structured software processes for the Web domain. Existing development environments and design methods provide abstractions from low-level implementation technology but lack generality and do not sufficiently address system maintenance and evolution.

*The WebComposition model* defines an object-oriented model for Web applications that abstracts from the Web implementation model and gives developers the power of object-oriented concepts for constructing reusable frameworks, for reuse by inheritance and delegation, and for improved modifiability and extensibility. However, a different approach to web application development is incorporating different approaches together, as developing any Web-based application completely through agile principles and practices can result in a delicate architecture and poor documentation, which makes the ongoing evolution of the system difficult.

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

International Journal of Innovative Research in Computer and Communication Engineering

*(An ISO 3297: 2007 Certified Organization)*

Vol. 4, Issue 5, May 2016

## REFERENCES

1. Knapp, N. Koch, G. Zhang, "Modeling the structure of web applications with argouwe", In proceedings of the 4th International Conference on Web Engineering, Springer, vol. 3140, pp. 771-72, 2004.
2. Athula Ginige, San Murugesan, "Web Engineering: An Introduction", IEEE, 2001.
3. Choudhari, Jitender, and Ugrasen Suman, "An Empirical Evaluation of Iterative Maintenance life Cycle Using XP", ACM SIGSOFT Software Engineering Notes 40, no.2, pp. 1-14, 2015.
4. Emilia Mendes, "Cost Estimation Techniques for Web Projects", IGI Publishing, Hershey, New York, 2008.
5. Hans W. Gellersen, Martin Gaedke, "Object- Oriented Web Application Development", IEEE Internet computing, vol. 3, no.1 pp. 60-68, 1999.
6. Somerville, Software engineering, 6th Edition, Addison-Wesley, 2000.
7. J. Conallen, "Web Applications with UML", Harlow, UK, Addison- Wesley Longman, 1999.
8. Jawadekar, W, Software Engineering: principles and practice, McGraw- Hill, New York, computer engineering series, 2004.
9. Murugesan, Yogesh deshpande, Steve Hansen, and Athula Ginige, "Web Engineering: A new discipline for development of web-based systems", In Web Engineering, pp 3-13, Springer Berlin Heidelberg, 2001.
10. N. Uikey, Ugrasen Suman, "A lifecycle Model for Web-based Application Development: Incorporating Agile and Plan-driven Methodology", International journal of Computer Applications, vol. 117, no. 19, May 2015.
11. Pressman R.S., "Software Engineering: A Practioner's Perspective", 5th ed., McGraw- Hill, New York, 2000, pp. 769-798.
12. Reifer, D.J., "Web development: estimating quick-to-market software", Software, IEEE, vol. 17, No. 6, pp. 57-64, Nov/Dec 2000.
13. S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera, "Designing data intensive Web Publications", Morgan Kaufmann Publishers, 2003.
14. Sukumar Letchmunam, "Pragmatic Cost Estimation for Web Applications", 2012.

## BIOGRAPHY

**Manpreet kaur** is a Research Scholar in I.K. Gujral Punjab Technical University, Jalandhar. She received M.Tech (Comp. Sc. And Engg.) degree in 2010 from Guru Nanak Dev University, Punjab, India. Her research interests are Software Engineering, Effort Estimation of software development and Web-based Applications.