A Lifecycle Model for Web-based Application Development: Incorporating Agile and Plan-driven Methodology

Nitin Uikey
Software Engineer
School of Computer Science & IT
DAVV, Indore

Ugrasen Suman
Professor
School of Computer Science & IT
DAVV, Indore

ABSTRACT

For conventional software development, generic software engineering lifecycle model has proven to be very important. Though, with the evolution of Web-based applications and internet, conventional software engineering models have limited support for developing Web-based applications. In recent years Web-based applications have become more complex and new technologies are emerging at a rapid pace. Therefore, the conventional software engineering lifecycle models need to be reformed in such a way that handling the change requirements and complexity of Web-based development becomes convenient for conventional developers. However, there is a lack of any generic process model available for Web-based applications. The paper identifies and analyzes various aspects of conventional and Web-based development and proposes a lifecycle model, which incorporates the aspects of agile and plan-driven development to develop a Web-based application successfully.

General Terms

Web-based Application Development Lifecycle, Agile, Plan-driven.

Keywords

Agile methodologies, Conventional software applications, Lifecycle Model, Feature Driven Development, Requirement Engineering, Scrum.

1. INTRODUCTION

The approach to software development plays a significant role in deciding the quality of software being delivered. The prime objective of software development industry is to deliver high-quality software. Non-planned and non-systematic approach to software development, if applied for complex software requirements, will certainly result in the development of low-quality and high-cost software products. This leads software professionals and practitioners to develop and deploy various software development lifecycles that can be successfully applied to various projects. With the advancement of time, the World Wide Web (WWW) has become the prevailing platform for universally accessible information and applications of any kind and here is where the web based applications are coming into play.

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. Therefor in immense preassure development of applications are becoming a challenging task [1]. 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 [2].

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 a shorter product life cycles and require continuous maintenance [3]. Various software process frameworks for web application development have been introduced, but they require a lot of resources such as developers, tools and equipment, Quality Assurance (QA) team and so on [4]. In Web-based application development, a high percentage of small software companies have been involved with approximately 10 to 50 employees working on a project [5]. The problems faced by these organizations are limited resource, limited skilled developers, lack of well-defined development methods and limited adopted quality management. Also, small software companies cannot afford the cost and risks of applying a rigorous software development process with limited development and management resources.Over the past five years, many software engineers have become concerned about the way Web-based applications are being developed [6] [7]. The concerns is some what related to poor suitability of traditional software engineering approaches and techniques to the development of Web-based systems.

Many conventional development methods have been proposed for constructing web applications such as, waterfall and spiral. However, these development methods are not adequate for developing web applications in small software firms [8] [9]. Consequently, agile development methods have been proposed to manage the problems that arise in conventional development approaches [10] [11]. The most popular agile methods used for software development for small teams and projects are Extreme Programming (XP) and Scrum [12]. However, the existing agile development methods have some limitations. These methods are found to be lacking in applying the important development practices as well as applying measurement practices during the development process [13] [14].

As most of the critical applications are migrating to the Web, the conventional approach has become increasingly infeasible for the development of Web-based applications. The current approach often fails or is facing the possibility of major failures due to the strong temptations to just “build it now and fix it later” to meet the deadline [15]. Existing literature shows that the main causes of Web-based applications development failures are unorganised development processes, flawed designs and poor management of development efforts[16]. In the absence of a systematic model for Web-based application development, the developers and organizations may face

serious problems in their successful implementation and maintenance. [17].

Through the literature it has become clear that a systematic development process needs to be followed for robust Web-based applications development. Also, the construction and evolution of Web-based applications require support smilar to conventional software applications through a well defined lifecycle process models. Therefore, there is a requirement for disciplined approaches and new methods, in order to avoid a possible Web crisis and achieve success in development and applications of complex and extensive Web-based systems [8].

This paper focus on conventional software development processes and how can the phases of various methodologies be put together to facilitate small software industries develop a quality Web-based applications. The rest of the paper is structured as follows. Section 2 analyze various conventional software development methodologies focusing on traditional and modern development methodologies. Section 3 evaluates the use of conventional development methodologies for Web-based applications. Based on the literature, analysis and requirements for an effective approach, section 4 proposes a lifecycle model for effective Web-based application development. The uniqueness of the model is incorporation of agile and plan-driven approach, and how effectively it suits the development process. Section 5 discusses the proposed model with an example and conclusion of the paper is presented in section 6.

2. EXISTING SOFTWARE DEVELOPMENT METHODOLOGIES

A software process model is a conceptual representation of a process [18] [19]. It presents a description of a process from some particular viewpoint. Correct and formal descriptions of software lifecycle activities can be developed using such models. Software project utilizes a process to facilitate execution of the engineering tasks and develop a product that satisfies the user requirements. As conventional software systems become big and complex, a number of development lifecycle models have been created to manage the process [20].

Waterfall model is the most influential and commonly used process model, in which the sequential execution of various phases of requirements specification, design, implementation, verification, and maintenance are performe. The Waterfall model has some well-known limitations as it assumes, the requirements are stable and well-known at the start of the project. Also, each phase relies on the preceding phase that needs to be completed before moving on. As requirements change are inevitable, with frequently changing requirements the approach results to inflexibility [21].

Bohem's model known as spiral model overcomes the limitations of the waterfall model [22]. Spiral model have four phases: Planning, Evaluation, Risk Analysis, and Engineering. These four phases are iteratively followed in sequence. However, the limitation of the spiral model is that, highly skilled people are required and process is more time-consuming and expensive [19] [22].The spiral model is based on evolutionary development. At first features with highest priority are defined and implemented. and then feedback from users is monitored. As the system evolves features with lower priorities are defined and implemented [23].

Agile approach effectively deals with changing requirements, which is difficult to manage in waterfall model. In Agile approach development activities are carried out in small phases, based on collaboration, adaptive planning, early delivery, continuous improvement, regular customer feedback, frequent redesign resulting in development of software increments being delivered in successive iterations in response to the ever-changing customer requirements [24]. Agile methodologies are increasingly being adopted by companies worldwide to meet increased software complexity and evolving user demands. The Agile software development embodies several methodologies including Extreme Programming, Scrum, Kanban, Lean, FDD (Feature-Driven Development), Crystal, DSDM (Dynamic Systems Development Method). However, agile is quite often used by small software developing teams, the required intensive customer involvement is difficult to achieve in practice for Web projects in small companies.

3. EVALUATING EXISTING METHODOLOGIES FOR WEB-BASED 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" [25]. 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 [26]. Gellersen and Gaedke defined Web-based applications as any software application that depends on the Web for its correct execution [27]. Koch and Kraus defined Web-based applications as "Web information systems that tend to be used to integrate and streamline business processes across organisations (customers, agents, suppliers, others) and geographical borders" [28]. 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.

A We-based application is invoked by a client (mostly by the Web browser) over the Internet, Intranet or Extranet. A Web-based application allows the information processing functions to be initiated remotely from a client (browser) and executed partly on a Web server, application server and/or database server [29]. These applications are specifically designed to be executed in a Web-based environment. Web applications are more complicated as compared to simple static Websites and provide a new way to deploy software applications to the end users. Web-based applications are based on a combination between art and technology [30]. Also, Web applications are interactive software which has complex Graphical User Interfaces (GUIs) and where numbers of back-end software components are integrated. These applications have revolutionized the business arena and have provided new opportunities to businesses and to the end users. The characteristics of Web-based applications can be broadly considered into following issues as shown in Table 1. Certain issues have been identified where Web-based applications differ from conventional software applications. For example, one characteristics of Web-based application is network intensive and it belongs to architectural issue of the software development. Network intensive implies that since Web applications are delivered to a diverse community of users, the applications is deployed on network architecture.

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.

1. **Informational:** information is provided to the end users through different and simple navigations and links.
2. **Interactive:** provide mutual interaction and communication among various users.
3. **Transactional:** users can request or place orders to obtain goods or services
4. **Form-based:** users can submit their data or queries to the organization and extract required information.
5. **Web Services:** it allows the user to create an interoperable distributed applications.
6. **Online marketplaces:** user can view various goods, can compare and purchase accordingly.
7. **Web Portals:** provides facilities to the users to other content of the web or services which are not part of the application.

Presently, Web-based applications have become much more complicated and extensive. The evolution of hand held communication and computing gadgets have come up with new challenges for developers and organizations. It is argued that mobile web-enabled services place unprecedented strains on the server infrastructure of the content provider and proposed resource management strategies to reduce the response time [31].

3.1 Web-based and other Applications

There is a growing body of research that is attempting to understand the differences between Web-based systems and conventional software systems [32]. Conventional applications consist only of 1 to 3 tier architecture, which resides on the clients machines, but Web-based applications resides on an n-tiered architecture. The Web-based applications are different from the conventional software in many ways. In general, a differentiation is made between the unique characteristics of Internet-enabled systems that are technical and organizational.

3.1.1 Technical Differences

The technical differences between Web-based applications and conventional software systems are obvious. It relates to the specific technologies that are used and how these impact on the structure of the application. The major of these are as follows: [33]

- **Use of specific technologies:** In Web-based applications, the link of business architecture and technical design is much bonded and information architecture is more complex than for conventional software systems.
- **Architecture:** Web-based applications have modularized architecture. Web applications are based on numerous Commercial off-the-shelf (COTS) components which are integrated with each other. Unlike conventional software where client and server have predefined characteristics and are static, Web-based applications the contents are generated dynamically.
- **Evolution of technologies:** The technologies of Web applications is evolving rapidly, the understanding of these new technologies is very crucial and restricted for developers who increase the project risks.
- **Compatibility and interoperability:** Web-based applications are often affected by some challenging factors that are the major sources for interoperability and compatibility issues. The servers might have various operating systems and the client might have different browsers. This bring challenges to developer of Web-based applications.
- **Content Management:** Content of Web applications are very crucial therefore, effective information design and suitable content management is important in these applications.
- **User Interface:** For conventional software application, users make an investment, buy it and install it on his/her machine and learn to use it. On the other hand, in Web-based applications user can switch from one Website to another competitor's Website with minimum effort.
- **Maintenance:** Web-based applications require frequent maintenance requirements than conventional applications.

3.1.2 Organizational Differences

With technical differences, there are also numbers of important organizational characteristics which are different in Web applications. The organizational point of view relates to the ways in which organizations make use of these systems. These important differences are as follows: [33].

- **Vision Driven:** Web-based applications are produced by a vision of the customer instead of need driven conventional applications.
- **Use of Design Tools:** In Web-based applications the business requirements of clients are changed swiftly therefore more effective design tools are required in these applications.
- **Product Delivery:** Mostly Web-based projects have a very short delivery schedule as compared to other conventional software projects.
- **Aesthetic Management:** In Web-based applications there is a continuous process of content updating and other interface changes and editorial changes.

Apart from technical and organizational differences, the comparison can also be performed with respect to development teams, requirements, contents, users, stakeholders, database integrity, level of abstraction and so on.

- **Multi-disciplinary development team:** Web-based applications require the collaboration of larger teams of people with diverse expertise compared to conventional software applications.
- **Diverse requirements:** Apart from the functional and non-functional requirements in the conventional software applications, navigational, content, personalization, structuring, access, aesthetic, marketing, database design, network security, and network performance requirements are also needed to be considered in the Web-based applications.
- **Substantial content:** Web-based applications contain text, audio, images and multimedia, so it is important to maximize the effectiveness and efficiency of managing content presentation.
- **Technology visibility:** Due to the distributed nature of Web-based applications link between the business architecture and the technical design of the system may be much bonded than for conventional software applications, and errors and downtime in Web-based applications are often not tolerated.
- **Diverse users:** Web-based applications are used by many users with various background and experience across the internet.
- **Multiple stakeholders:** Unlike conventional software applications, a diverse range of users, people who maintain web systems, the organization using web systems, and system developers are involved.
- **Short development cycle:** Because of the dynamic business environment, fast changing Web technologies and users/customer's requirements, Web application projects usually have shorter development cycles required than conventional software applications.

Web-based application development is neither a clone nor a subset of software engineering, although both involve programming and software development. While Web-based applications use software engineering principles, it encompasses new approaches, methodologies, tools, techniques, and guidelines to meet the unique requirements of Web-based applications. The difference between Web-based applications and Traditional software applications can be summarized as shown in Table 2.

Table 2. Difference between Traditional and Web-based applications

Traditional software applications	**Web-based applications**
Software system has small user range	Web-based applications have large user range
User Requirements are specific	User Requirements changes with time
Growth and change are small	Rapidly changing
Development budgets vary in a wide range according to the size of the company	Development budgets are small.
Development time is longer	Development time is small.
Hardware and Software environments constraints are specific.	Hardware and Software environments constraints are not specific
Less emphasis on user interfaces	More emphasis on user interfaces.

3.4 Need for a lifecycle model for development of Web-based applications

There have been numerous attempts to identify the important success factors and to use engineering approaches to building process models to address specific concerns [34]. The implementation model that the Web is based on makes it difficult to apply existing process models to the development and evolution of Web-based applications [17]. As identified in above section, there is significant evidence that Web-based applications development has particular characteristics that differentiate them from traditional software application development.

Waterfall is a more sequential approach to development. Upfront planning is required to ensure requirements and project details are manged properly and no major requirements arise in between development. This results in clear project understanding among developers and customers. On efficient requirement management, prject can be launched faster and project budgets can be estimated more accurately. However, when dealing with a business that continually evolves and requirements frequently change, the waterfall appraoch can lead to scope creep, budget overruns, or a project launch that doesn't meet client's requirements. Re-executing a waterfall process iteratively will eventually result in wastage of resources and cost over-run.

Agile methodology, on the other hand, enable team and customer coordination, produce products in smaller delivery sets. In Scrum, delivery sets are the prioritized feature/task list producted in a sprint. Scrum Master ensure the team coordination and collaboration. Daily standup meething are small duration meething for status check to identify the status of the developers and resolving the problems faced in previous days development. Sprint is the phase where actual deelopement occur. When dealing with a dynamic business, where evolution is continuous and requirements may change in between development, the agile process can create a never ending loop. The final product can often differ from what was originally planned resulting in overrun budget expectations.

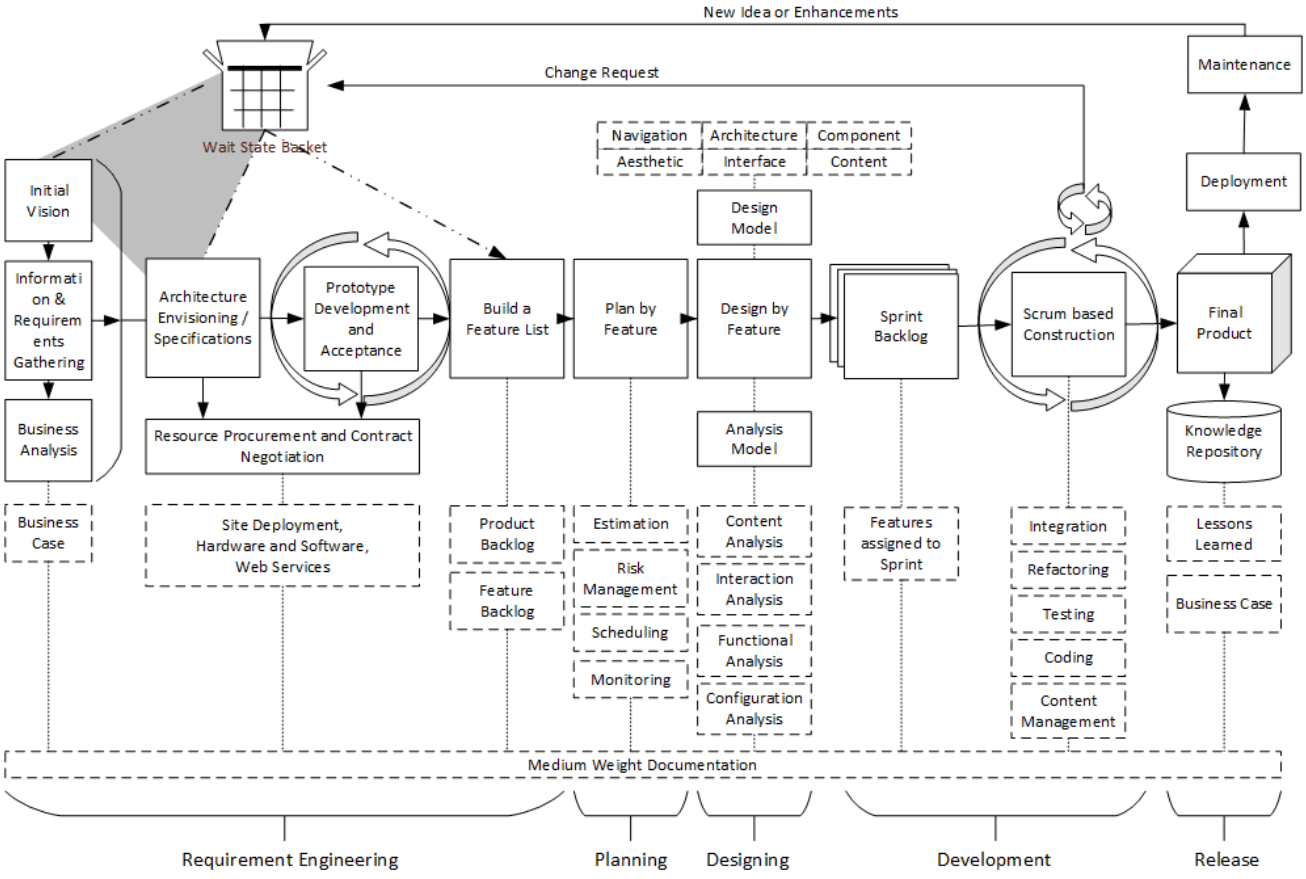

Figure 1. A lifecycle model for Web-based application development

Also, 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.

It is said that Web-based application development is not a perfect "clone" of software engineering, but it borrowed many software engineering's fundamental concepts and principles, emphasizing the same technical and management activities [35]. Therefore, this paper considers that a lifecycle model approach to Web-based application development can be proposed incoporating various aspects of agile and plan-driven approach to meet various unique characteristics of Web-based applications.

4. PROPOSED LIFECYCLE MODEL FOR WEB-BASED APPLICATIONS DEVELOPMENT

Ensuring a systematic yet flexible lifecycle model to be used in small software industries, the proposed lifecycle model is based on various software engineering fundamentals. Here the development teams are encouraged to adapt and improve their working practices iteratively and in a sequential manner, where tasks are preceded by the completion of previous tasks.

Figure 1 presents a lifecycle model, which incorporates 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 are sequential in nature. These phases are further defined in the subsequent subsections.

4.1. 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. Basically it is the first broad aspect of the customer's need. After understanding the initial vision of the customer, the project manager proceeds for requirement and information gathering. The information to be collected from the customer can be based on some questionnaire, interviews, business case, and so on. The requirements can be based on functionality, performance and usability, so that the development and final product environment can be created. This facilitates the manager to create a picture of the product and establish a goal. 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.

4.2 Architecture Envisioning

Being a part of agile modelling, with architectural envisioning the team perform some high-level architectural modelling early in the project to help foster agreement regarding the technical strategy within the team and with 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 (agilemodelling.com).

4.3 Prototype Development and Acceptance

Based on the user requirements and architecture envisioning, the team is able to create a prototype of the system. As we are dealing with Web-based application development, the prototype can be more focused on broad view of the entire system. Focus is more on user interaction than low-level system functionality, such as database access. Benefits of creating a 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.

4.4 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. Once a contract is in place, the relationship between the organization and the provider needs to be managed to ensure that the work proceeds according to plan. While the acquisition of internal resources may term to a service-level agreement and can be negotiated with the customer.

4.5 Feature Driven Development (FDD)

In the proposed lifecycle model, the next three phase after prototype development and acceptance are derived from FDD. The phases are built a feature list, plan by feature and design by feature. FDD stresses in creating shorter iterations of functionality, with each functionality catering to certain features on the website. In FDD, the software designing and modelling is given due importance, apart from tracking and report options as well, making it quite ideal for deployment in corporate websites. It gives more importance to the basic aspects of development, that when neglected, pose great problems as the development process progresses. It is a comprehension of many organizational best practice methodologies used. 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.

4.6 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. In the proposed lifecycle model, 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. Scrum has a very clear project management emphasis. Scrum is predicated on the concept that software development is not a clearly defined process, but a series of phases with complex input/output transformations. 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 centres 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. The conclusion of each 30 day Sprint involves the software demonstration of the product features that have been completed during that Sprin [37].

4.7 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. Maintenance comprises of lightweight iterative processes based on user's/customer's demands and may go many times but will not fundamentally change the Web-based application [36]. 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. As the goal is to get launchable product deployed as quickly as possible to get revenue generation activities initiated.

5. DISCUSSION

The focus of the paper is to examine an insight of traditional software development processes and how can the phases of various methodologies be positioned together to facilitate small software industries develop a quality Web-based applications. As the approach is conceptual, a software development case study should help in explaining the model.

The proposed model provides a systematic approach to help in understanding and guiding Web-based applications development incorporating agile and plan-driven methods. The model identifies a set of requirement engineering, planning, designing, development and release, along with evolution, knowledge repository and maintenance. This study identifies mix and match approach as an essential mechanism to address challenges faced in project development and develops a model that describes the way to develop a Web-based applications. Practitioners can use the model as shown in Figure 1, to guide their development efforts.

The unique characteristic of the proposed model is the ability to harness various features of agile and plan-driven models to satisfy the required development process of Web-based development. As Web-based application development is different from conventional software development, it requires mix and match type of approach which can fully satisfy the

development needs. The proposed model contains sequential approach of plan-driven, where the succeeding phase will start after the current phase is completed. The model also uses approaches of agile methodologies. Parts of FDD assists in completing analysis and designing, whereas, for actual development, the concept of SCRUM with some practices of XP is used. The components of the model can easily be explained through an example.

5.1 Requirement Engineering Process

For instance, there is a customer with an initial vision of creating an online solution for property rental business approaches a software development organization. The team collects an initial requirement of the customer and can be detailed through a case study of a similar kind. The requirements identified can be as follows.

The customer wants a system where the users can find a property to buy or rent. Enable registered users to locate agents who have properties to rent or sell. Agents can upload details of the property. The customer wants complete admin control of the system. Search facility should be available to users with detailed search based on various criteria The application should have a live chat support for users and agents.

Business analysis of the requirements is performed, which identifies the project as a client-server based, informational and transactional. Architecture envisioning succeeds the business analysis phase, with the aim to create a structure of the application and an architectural model based on the category of the application. The team now quickly works on the prototype of the application which enriches the customer with its look and feel. The presentation implies the template, colour scheme, layout and the structure of the forms or web pages visible to the user. The prototype is enhanced or modified until accepted by the customer. Simultaneously, another team can work out on resource procurement and contract negotiation phase. Resource procurement can be the amount of server space required, domain registration, technical issues for hosting web application and so on. It is important that resource procurement and contract negotiation can be performed only after architecture envisioning and prototyping phases are finalized. The next phase in requirement engineering is built a feature list. This phase is derived from FDD, a methodology from agile family. The team and the customer work collaboratively to finalize the features to be developed in the application. The feature list for the above identified requirements can be as follows.

- Develop a website with two types of users: Users – who need the property for rent or to purchase; Agent – who had various property for sale or rent.
- The website will allow agents to add a property with details pertaining to the property.
- An administrative section provided on the website will allow the customer to exercise total control over the various modules on the site.
- An advance search functionality will be included on the website which allows users to search for apartments based on city, state, apartment home features along with other policies.
- A live chat interface is another feature that will be added to allow registered users to ask for help or advice if needed. Defining access controls on each web page to allow only authorized registered users to access the listings.

5.2 Planning Process

In this phase, the team and the customer plan the order that the features are to be implemented, based on feature dependencies, load across the development team, and the complexity of the features to be implemented. A typical scenario is to consider the development sequence, the assignment of feature sets to the development team. The planning team estimates the duration, dependencies between features and resource required for the features to be completed. The team also keep in mind the planning of risk management and monitoring of the development. The team and the customer can prioritize the features based on the risk and can bring the high- risk and complex features first. Monitoring can be performed by establishing external milestone such as betas, previews, feedback checkpoints and so on. The planning is a team activity. Therefore self-assessment is achieved by the active participation of the team and the customer, who use the knowledge they gained from architecture envisioning and prototyping to help make better-informed decisions.

5.3 Designing Process

Designing phase comprises a per-feature activity to produce the features design package. The team develops the detailed sequence diagrams required for each feature being designed. The team writes up and records any alternative designs, design decisions, assumptions, requirements clarifications, and notes in the design alternatives or notes section of the design package. The developers refine the model to add additional classes, operations, and attributes based on the sequence diagrams defined for the features. The team conducts a design inspection. On acceptance, a to-do list is created, and each team member adds their tasks to their to-do list.

At modelling phase, analysis modelling and design model are also required. Analysis activities help in designing to understand the detailed requirements that will satisfy customer needs. Analysis model includes analysis of content, interaction, function, and configuration. The content analysis identifies the full range of content to be provided by the Web-based application, which includes text, graphics and images, and video and audio data. Interaction analysis describes the manner in which the user will interact with the application. The functional analysis defines the operations that will be applied to the content and describes other processing functions that are independent of content but necessary to the end user. Configuration analysis describes the environment and infrastructure in which the application is deployed. Aesthetic and navigation design are missing in the conventional software engineering process model. But, these are necessarily required for design model, as aesthetic design describes the look and feel of the application. A good appealing and attractive design of the application generates appropriate responses from users. It includes colors, layout, text size, font and placement, and so on. Navigation design represents the navigational flow between content and for all applications functions. Apart from conventional software navigation, in Web-based application navigating from one webpage or form to another and back again can result in confusion, if not designed effectively. Rest items of design model shown in the proposed model are an integral part of the project designing.

5.4 Development Process

On successful completion of the application designing and prioritizing the features list, the team can proceed with application development. The development process in the proposed model is derived from well-known agile

methodologies called SCRUM and XP. The development phases are taken from SCRUM as sprint backlog and SCRUM based construction. The prioritized features list build in designing becomes sprint backlog or the features assigned to the sprint. Here the team can choose from the backlog and choose the features to be developed in the sprint. The development is carried out in the sprint, which last for two to four weeks. Throughout the development, various activities are performed like coding, integration, testing, content management and refactoring. Web-based application development is more innovative in comparison to conventional software development, as developers need to keep in mind various elements of the designing process. For instance, in static web page development, a lone team member may create a module, but if the application module is more based on aesthetic and navigation model, the developers can follow the pair-programming method. Therefore, incorporating XP based practice as per the requirement of the development. The product constructed in the development process has to undergo various testing, of which integration and system testing plays an important role before the product is finally released to the market.

5.5 Release and Evolution Process

Once the product is constructed, it is released and deployed where the users can actually use the application. One of the reasons for the popularity of Web-based applications among developers is the ability to release a version without having to distribute and install the new version on client computers. The browser acts as a universal client, and the application exists only in one copy on the server. Following the deployment of the system, maintenance starts and it involves the modification of the system based on client's demands and feedback so as to correct defects and upgrade the system, which involves making small changes instead of major changes to the application. The major change or a new idea to be added to the application are kept in a wait state basket. From this basket, the team may start working on the idea as soon as the minor changes and modifications are complete. The team have to again start the lifecycle model till the ideas are finally deployed.

6. CONCLUSION

Presently, the automation of business organizations is majorly based on Web-based development. On the analysis of existing software development methodologies and literature review, it is observed that existing software development methods have limitation in developing Web-based applications. As Web-based applications are more evolutionary, changes to the application occur very frequently. Waterfall model being sequential in nature is unable to manage frequent changes. Spiral model on the other hand requires highly skilled developers and is time consuming. While, agile methods are capable to manage frequent change but developing a Web-based application through agile principles and practices can result in indefinite iterations. Further, the paper identifies technical and organizational differences between conventional and Web-based development. There exist several Web-based applications such as yahoo.com, facebook.com, amazon.com, rediff.com and so on. Such Web-based applications need strategic development for the effective execution of business and organization. In this paper we have analyzed existing and Web-based application development and proposed a lifecycle model for the development of Web-based applications. Although the proposed lifecycle process model includes characteristics from the conventional software development lifecycle process model, the combination and sequence of applying these characteristics are unique and, therefore, represent an innovative approach to Web-based applications development. The lifecycle process model is proposed to be a generic process for Web-based applications development, although a number of variants are expected in the future to suit different applications cases such as Mobile-based application development (Mobile Apps) and depending on the nature of projects.

7. REFERENCES

[1] Abdesselam Redouane," Guidelines for Improving the Development of Web-Based Applications" Proceedings of the Fourth International Workshop on Web Site Evolution (WSE'02) 0-7695-1804-4/02 2002 IEEE

[2] Said Hadjerrouit, "Web-based Application Development: A Software Engineering Approach ACM SIGCSE Bulletin June 2001 Vol 33. No. 2 p 31-34.

[3] Rodríguez, Daniel, Rachel Harrison, and Manoranjan Satpathy. "A generic model and tool support for assessing and improving Web processes." In Software Metrics, 2002. Proceedings. Eighth IEEE Symposium on, pp. 141-151. IEEE, 2002.

[4] Ahmed E. Hassan and Richard C. Holt "Migrating Web Frameworks Using Water Transformations", Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPSAC'03) 0730-3157/03 2003 IEEE.

[5] Laporte, Claude Y., A. Renault, J. M. Desharnais, N. Habra, M. Abou El Fattah, and J. C. Bamba. "Initiating software process improvement in small enterprises: Experiment with micro-evaluation framework." In SWDC-REK, International Conference on Software Development, University of Iceland, Reykjavik, Iceland, pp. 153-163. 2005.

[6] Coda, Francesco, Carlo Ghezzi, Giovanni Vigna, and Franca Garzotto. "Towards a software engineering approach to web site development." In Software Specification and Design, 1998. Proceedings. Ninth International Workshop on, pp. 8-17. IEEE, 1998.

[7] Pressman, R. S. (2000). What a tangled web we weave [web engineering]. Software, IEEE, 17(1), 18-21.

[8] Huang, Wei, Ru Li, Carsten Maple, Hongji Yang, David Foskett, and Vince Cleaver. "Web Application Development Lifecycle for Small Medium-Sized Enterprises (SMEs)(Short Paper)." In Quality Software, 2008. QSIC'08. The Eighth International Conference on, pp. 247-252. IEEE, 2008.

[9] Altarawneh, Haroon, and Asim El Shiekh. "A theoretical agile process framework for web applications development in small software firms." In Software Engineering Research, Management and Applications, 2008. SERA'08. Sixth International Conference on, pp. 125-132. IEEE, 2008.

[10] Stojanovic, Zoran, Ajantha Dahanayake, and Henk Sol. "Modeling and Architectural Design in Agile Development Methodologies." EMMSAD'03 (2003): 1-10.

[11] Lindstrom, Lowell, and Ron Jeffries. "Extreme programming and agile software development methodologies." Information systems management 21, no. 3 (2004): 41-52.

[12] Qumer, Asif, and Brian Henderson-Sellers. "An evaluation of the degree of agility in six agile methods and its applicability for method engineering." Information and Software Technology 50, no. 4 (2008): 280-295.

[13] Turk, D., France, R., Rumpe, B. (2002). Limitations of Agile Software Processes, Third International Conference on eXtreme Programming and Agile Processes in Software Engineering, Sardinia, Italy, 43-46.

[14] Jiang, Li, and Armin Eberlein. "Towards a framework for understanding the relationships between classical software engineering and agile methodologies." In Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral, pp. 9-14. ACM, 2008.

[15] Lowe, David. Hypermedia and the Web: an engineering approach. John Wiley & Sons, Inc., 1999.

[16] I Sommerville. Software Engineering, 6th Edition, Addison-Wesley, 2000.

[17] Murugesan, San, 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.

[18] Scacchi, Walt. "Process models in software engineering." Encyclopedia of software engineering (2001).

[19] Jalote, Pankaj, Aveejeet Palit, Priya Kurien, and V. T. Peethamber. "A Process Model for Iterative Software Development." Infosys Technologies Limited Electronics City, Bangalore-561 229 (2003).

[20] Kappel, Gerti, Elke Michlmayr, Birgit Pröll, Siegfried Reich, and Werner Retschitzegger. Web engineering–old wine in new bottles?. Springer Berlin Heidelberg, 2004.

[21] Pressman R.S., 'Software Engineering: A Practitioner's Perspective', 5th ed., McGraw- Hill, New York, 2000, pp. 769-798.

[22] Boehm B, 'A Spiral Model of Software Development and Enhancement', ACM SIGSOFT Software Engineering Notes, ACM, 11(4):14-24, August 1986.

[23] Jawadekar,W, Software Engineering: principles and practice, McGraw- Hill, New York, computer engineering series, 2004.

[24] Matharu, Gurpreet Singh, Anju Mishra, Harmeet Singh, and Priyanka Upadhyay. "Empirical Study of Agile Software Development Methodologies: A Comparative Analysis." ACM SIGSOFT Software Engineering Notes 40, no. 1 (2015): 1-6.

[25] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera. Designing Data-intensive Web Applications, Morgan Kaufmann Publishers, 2003.

[26] J. Conallen. BuildingWeb Applications with UML, Harlow, UK: Addison-Wesley Longman, 1999.

[27] H. W. Gellersen, M. Gaedke. Object oriented web application development. IEEE Internet Computing, vol. 3, no. 1, pp. 60–68, 1999.

[28] A. Knapp, N. Koch, G. Zhang. Modeling the structure of web applications with argouwe. In Proceedings of the 4th International Conference on Web Engineering, Lecture Notes in Computer Science, Spinger, vol. 3140, pp. 771–72, 2004.

[29] Finkelstein, Anthony CW, Gerti Kappel, and Werner Retschitzegger. Ubiquitous web application development-a framework for understanding. 6th World Multiconference on Systemics, Cybernetics and Informatics, Orlando, Florida, US. 2002.

[30] Powell, Thomas A., David L. Jones, and Dominique C. Cutts. Web site engineering: beyond Web page design. Prentice-Hall, Inc., 1998.

[31] C. Canali, M. Colajanni, R. Lancellotti. Resource management strategies for mobile web-based services. In Proceedings of IEEE International Conference on Wireless and Mobile Computing, Networking & Communication, pp. 172–177, 2008.

[32] S. Overmyer, "What's different about requirements engineering for web sites?" Requirements Engineering Journal, vol. 5, no. 1, pp. 62–65, 2000.

[33] Lowe, David, and Brian Henderson-Sellers. "Characteristics of web development processes." SSGRR-2001: Infrastructure for E-Business, E-Education, and E-Science (2001).

[34] H. W. Gellersen, M. Gaedke. Object oriented web application development. IEEE Internet Computing, vol. 3, no. 1, pp. 60–68, 1999.

[35] R. S. Pressman. Can internet-based applications be engineered? IEEE Software, vol. 15, no. 5, pp. 104–109, 1998.

[36] Choudhari, Jitender, and Ugrasen Suman. "An Empirical Evaluation of Iterative Maintenance Life Cycle Using XP." ACM SIGSOFT Software Engineering Notes 40, no. 2 (2015): 1-14.

[37] Clutterbuck, Peter, Terry Rowlands, and Owen Seamons. "A case study of SME web application development effectiveness via Agile methods." The Electronic Journal Information Systems Evaluation 12, no. 1 (2009): 13-26.