CHAPTER 1

INTRODUCTION

Introduction

1.1 Software Applications vs. Web Applications

An application running on a single machine and benefiting its user only i.e., being used for solo purpose, constitute software application and an application running on a network, providing services to its users on the network itself, is called web application. Popularity of web application has risen exponentially in last decade and every day numbers of users of web applications are increasing very rapidly. Table 1 represents a very brief idea of both types of applications in terms of similarities and advantages.

Brief Introduction	***Software Applications***	***Web Applications***
Similarities	• Both applications are computer programmes installed at some computer, software applications being installed at user's computer and web applications being installed at server's computer.	
Dissimilarities	• Installation takes place on user's side. • No need of network uses for running the software application on the computer. • Performance of software applications depends solely on user's computer's hardware configuration. • Software applications are purpose specific. • In order to add some new functionality, a new version is needed to be released.	• Installation takes place on server's side, which is centralised computer. • Server provides services to its users via internet. • Performance of web applications largely depends upon network's configurations along with hardware configurations. • Web applications are multipurpose service providers. • At any time new functionality can be added to web application without attention of user.

Advantages of Software Applications over Web Applications	• No dependability on internet. • No limit of complexity of software applications. The only requirement is good hardware configuration. • Security from viruses and unauthorised users is needed to be provided on user's level only.	• Availability of internet is an unavoidable issue along with its data transfer efficiency. • Network speed limits complexity of web applications by a very huge amount. • Security needs to be provided on network level.
Advantages of Web Applications over Software Applications	• Needs complete installation at user's side thus requiring a large memory space. • If any major problem or an important requirement is found and need to be fixed/ added, new version of the software must have to be installed. • Back up of user's data have to be maintained by user himself. • Special efforts need to be made for making software application compatible with various operating platforms.	• Application is installed at server's computer and services are provided via network. Thus user needs not to install application on his/her computer. • Any update or fixing is made online without attention of user. • Data is stored at server's side which is kept safe and backed up with zero user consideration. • Most web applications are much far compatible on distinct platforms, even they can be accessed on mobiles phones.

Table 1 Brief Introduction of Software Applications and Web Applications

1.1.1 Software Applications

Software application is all the computer software that causes a computer to perform useful tasks benefiting its users beyond the running of the computer itself, thus, making it distinct from system software which manages computer capabilities but does not perform users' tasks. Thus software application exploits the benefits of some particular computing platform and works for some particular purpose. Some applications have several versions in order to run on different computing platforms. Sometimes, a new application, made for working on a single platform, becomes so popular that it increases desirability of that particular computing platform. Such applications are known as *killer applications*.

Software application may be classified in two general categories - horizontal application, and vertical application. *Horizontal applications* are most popular and widespread software package designed to be used by many different kinds of departments or companies, such as office automation or bookkeeping software. *Vertical applications* are niche applications developed for being used by some specific user domain such as insurance billing software application which can be used by only some specific insurance department.

There are many types of software application based on their broad working type:

- *Application Suite* - Multiple applications bundles together form application suite. They may have related functions, or may interact with other, even may be authorized to open each other's files. Example - Microsoft office, OpenOffice.org.
- *Enterprise Software* - These deals with the needs of organization process, probably in a distributed manner. Departmental software is subset of enterprise software which addresses the need of small organization or a part within organization itself. For example Customer Relationship Management (CRM).
- *Information Worker Software* - Word processor, personal information system, spreadsheets etc. are examples of such type of software letting users create and manage information.
- *Content Access Software* - Media players, web browsers etc. are content access software which allow to access content without editing it. These softwares fulfill the need of individuals and groups to use digital entertainment and published content.
- *Educational Software* - These are also content access software but the content is to be used by educators or students. For example, mark sheet generation software.
- *Simulation software* - Such type of software is used to simulate physical or abstract systems for the purpose of research, training or entertainment.
- *Media Development* - Media development software produces electronic media for others to consume and is most often used in a commercial or educational field. Examples are graphic-

art software, desktop publishing software, multimedia development software, HTML editors, digital-animation editors, digital audio and video composition, and many others [2].

- *Product Engineering Software* - Computer Aided Design (CAD), Computer aided engineering (CAE) are product engineering software being used in developing hardware and software products.

1.1.2 Web Applications

Web application is an application that is used by various types of users over a network which may be intra-network or inter-network. Web application uses a web browser as a client which request/post some data from/on server which again may be centralized or distributed in order to improve server response time. Client is a computer software application installed at users' side which runs web application, developed in a browser supporting language (such as javascript combined with HTML). Figure 1 represents a very generic view of web application.

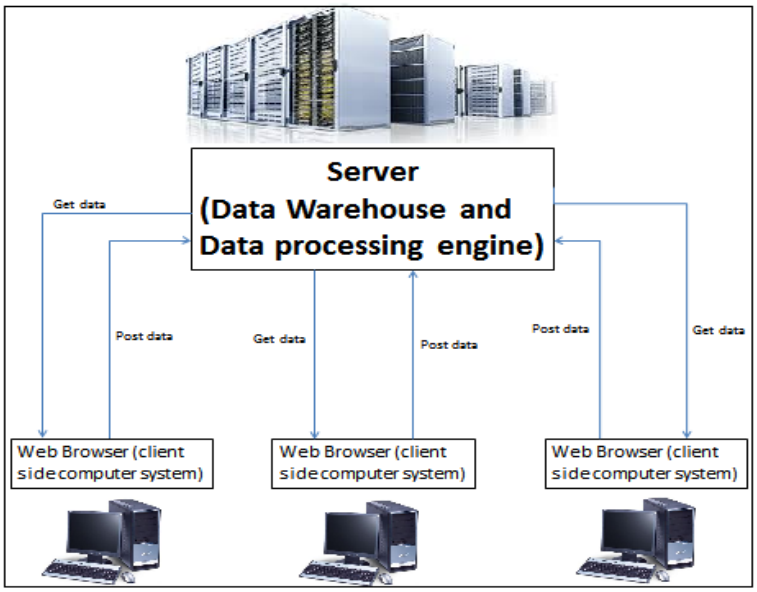

Figure 1 Generic View of Understanding of Web Application

1.1.2.1 History

In early days, web pages used to be static but sequence of pages could provide an interactive experience, and user input was returned through web form elements embedded in the page mark-up. In 1995, Netscape introduced JavaScript, a client side scripting language, which allows addition of dynamic content in the webpage running on the client side. In 1996, macromedia introduced Flash to embed animation in the web page. In 1999, the "web application" concept was introduced in the Java language in the Servlet Specification version 2.2 [19]. In 2005, Ajax came into

existence and application like Gmail started to become more interactive at client side. A web page script is able to contact the server for storing/retrieving data without downloading an entire web page [51]. In 2011, after introduction of HTML 5, applications became capable of graphic and multimedia without using client side plugins.

Here are some types of web applications:

- *Analysis Campaign* - Create advertisements and track them in order to produce the results for analysis of the product. Analysis report may exist in form of various types of graphs and charts.
- *Order Goods Online* - There are many websites available for online purchasing like flipkart, snapdeal, myntra, amazon etc.
- *Estimate* - Analysis of user behavior is performed on various sites in order to get current user trends and taking business decisions accordingly. Example - Rossiter and Co – video estimate.
- *Educational Sites* - There are many sites running successfully providing educational material for students of various subjects. Example - w3schools, APSA etc.
- *NEWS Sites* - This is very popular and useful kind of websites providing information of happenings across the whole world. Each news channel has its own website telecasting their news bulletin over internet. Example - BBC news, NDTV etc.
- *Searching Sites* - Google, yahoo, bing etc. are some well-known name when one desires to search on internet regarding some specific website.
- *Travelling Sites* - These sites provide information about various convince like trains, buses, flights etc., also gives facility to book ticket online. Example - makemytrip, IRCTC etc.
- *Social Sites* - These sites provide a platform for internet users to connect with each other sharing their information relating to the purpose of the website. For example, *Facebook* provides users a way to share their thoughts with each other.

1.1.3 Advantages and Disadvantages of Web Application over Software Application

Using web application rather than desktop application is advantageous due to various reasons listed below.

- *Zero install* - To run web application user need only a web browser installed on his computer machine; no additional software application required to run web application. Example - say, we have to write some document and transfer it to someone. One case is using Microsoft office desktop application which requires to be installed on computer; other case is using

online version of document editor like “google doc” for which we need not to install any application on the computer.

- *Cross platform compatibility* - Most web applications are much far compatible than software applications on various platforms like windows, linux, iOS, and even on mobiles though there may be need to install different versions of web browser for different platforms.
- *Reduce business costs* - Web applications provide effective communication between employees along with eliminating need of printed material, and allowing employees to update their work more easily and rapidly.
- *Application is in one place* - Web application is installed at server, thus making updating process much easier and error free. Also, accessing the application from anywhere is possible without need of any special software installation requirement and still if any, it may be downloaded from the server or some other place easily.
- *Availability of web servers* - Accessing web application is available 24 hours, 7 days a week. There is no accessing time limitation until constrained intentionally.
- *Data stored on the server is safe* - Storing data on personal computer is less safe than storing on the web server as various security software and protocols are run on the server with daily data backup facility. In case, personal data storage device gets crashed, data can be retrieved from the server.

There are some limitations and drawbacks also after having so much advantages of using web application over software application which are listed below.

- *Slower speed of internet* - The main limitation of working on web application is transmission speed of media used in setting internet network. Some websites having high user transactions at a time takes some time to get load, and sometimes user request gets vanished in between so request has to be sent again and again. Some websites do not run on low speed internet.
- *Various firewalls* - There is another problem which is due to firewalls applied into some networks thus prohibiting access of some specific websites.
- *Security risks* - though data stored on server is safer but transmission of data is never satisfyingly secure over internet. There is a threat caused by hacker and viruses meant to harm or steal the useful information.
- *Availability of internet* - Though servers may be available most of the times but availability of internet facility is doubtful as user need to make some arrangements to make sure internet’s availability.

- *Limited transfer limit of data* - Due to limited transmission rate of data over internet, amount of data transferred against time taken has to be limited depending upon time availability constraints of user.

1.1.4 Similarities and Dissimilarities between Web Application and Software Application

Software application and web application, both are computer programs, where software application is installed on users' machine itself and web application is installed at server which is centralised machine with users having a client program for accessing the computer application installed at the server.

One major difference in terms of users is that software application is generally specific to some purpose and thus having limited type of users but in case of web application concept of diverse users comes into scene making requirement specification phase of application development complex.

Another advantage with software application is that once installed on users' machine, it is up to that specific user owing the machine that how he uses the software and whether he uses updated version or the older one. It does not affect any other users' working experience using the same software on his owned machine. In case of web application, a lot of things need to be considered in terms of security (from hackers) and also unintentional bad data input by the user.

What comes out as conclusion is that if the software is installed at some centralised machine and users have some host program installed at their machines to use the software through network simultaneously then it will be put in list of web applications otherwise desktop application.

1.2 Motivation

A Web-based application refers to any computer program that is accessed over a network using HTTP connection, rather than existing within a device's memory (unlike desktop application running on desktop without need of any network). Sir Timothy John made a proposal for information management system in March 1989 giving birth to web. Web was released into public in 1993 as web 1.0 containing only static information on the web page and from that day, popularity of web has been increasing so intensively that the Indexed web contains at least 14.78 billion pages (Thursday, 09 May, 2013) [46]. Web 2.0, *the social web*, introducing dynamic web pages and interactive user interface, rose in 2006. Advanced searching (graphical, dialogue, and natural language searching), blogging, concept/ mind mapping, mash ups, podcasts, RSS feeds, bookmarking, social networking, web conferencing, wikis etc. are examples of web 2.0 [3]. By 2016, web 3.0, *the semantic web*, is expected to come into scene after revolutionary acts by web 2.0. This

fastest increasing popularity has been demanding newer technologies for developing web applications more robust, faster, and user interactive.

1.3 Guided Tour

This research paper enlists various technologies for developing web applications classified according to their development approach along with difference between development strategies of web applications and software applications.

Chapter 1 explains software applications and web applications defining similarities and dissimilarities between them, and also advantages and disadvantages of using web applications over software applications.

Chapter 2 presents related work proposed by various researchers for developing web applications.

Chapter 3 defines development methodology and its importance in developing the application along with difference between software application development methodologies and web application development methodologies.

Chapter 4 gives overview of various types of approaches for web application development. The list includes agile development approach, Object Oriented development approach, UML based development approach, Sequential development approach, Reverse Engineering based development approach, and Knowledge based development approach.

Chapter 5 describes web application development methodologies proposed by various researchers based on the approaches defined in Chapter 4. Total ten methodologies have been explained briefly in Chapter 5.

This thesis work includes three Appendixes. Appendix A and Appendix B present development of a web site *Red Drop* using XP, Agile approach based methodology, and Sequential and UML based methodology respectively. *Red Drop* is an online blood donation website; Appendix C presents some screenshots of the same.

CHAPTER 2

RELATED WORK

Related Work

Since introduction of web into public as *Web 1.0* in 1993, web development techniques have been evolved and raised in exponentially increasing count. The work in this thesis specifies 10 web development methodologies based on different approaches. The very first in this series is *Web Application Development using CORBA* (Common Object Request Broker Architecture) presented by Eun Sook Cho et al. in 1997 at IEEE [9]. This is an object oriented web application development approach that brought phenomenally rapid expansion in World Wide Web. CORBA is popular because of service suits it provides including services like naming, security, licensing etc.

Second in the series is again object oriented technique for developing web applications presented by Martin Gaedke et al. in IEEE Internet Computing in 1999 named *WebComposition* [50]. WebCommpostion technique uses two modules: Component Model and Markup language described in Chapter 5. Main concept of the technique is modelling web entities like web pages or some specific web link at arbitrary level of abstraction and presenting specifications in WCML (Web Composition Markup Language) which is based on XML.

Extreme Programming (XP) is next approach based on agile development principles to develop web applications. Frank Maurer et al. presented a paper in IEEE Computing in 2002 named *Extreme Programming: Rapid Development for Web based Applications* specifying 12 agile practices to be followed while developing web applications [11]. Appendix A comprises of the case study of website development using the same.

Next in the series is reverse engineering based development technique presented by Filippo Ricca in IEEE Computer Society in 2004 by name *Analysis, Testing and Restructuring of Web Applications* [10]. In this technique, new models for developing web applications are derived from existing models following three step process Analysis, Testing, and Re-structuring.

In 2005, Makoto et al. presented a research paper *Knowledge Based Experimental Development of Web Application Systems* at Proceedings of the 2005 The Fifth International Conference on Computer and Information Technology (CIT'05) organised by IEEE [22]. They proposed a software KITS (Knowledge based Integrated quesTionnaire Software) consisting of three sub-systems, the questionnaire subsystem, the software product line subsystem and the knowledge database subsystem for developing web applications.

The list of web application development methodologies continues with UML based approach *Web Application Development by Supporting Process Execution and Extended UML Model* proposed by Wookjin Lee et al. presented at Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSCE'05) organised IEEE in 2005 [55]. The development methodology is based on agile

development approach consisting of navigation model and component communication model, extended from state machine package and interaction package of UML 2.0 respectively.

Next development methodologies described in the thesis is *Agile Web Development with Web Framework* proposed by Hu Ran et al. in 2008 [17]. AWDWF emphasize on communication between people for example, the communication among the development team members, and between the customer and development teams. In order to handle rapidly changing customer requirements, prototypes are released frequently and this helps in quick and timely customer feedback. The main idea behind AWDWF is keeping development team focused on the core business logic and thus avoiding programming duplication and wastage of resources, consecutively, improving system's efficiency and stability, and maintaining quality of the web application.

In 2008, Wei Huang et al. proposed *Web Application Development Life Cycle for Small Medium-sized Enterprises (SMEs)* [53]. This research paper, published in IEEE in 2008, proposed a mixture of sequential and iterative approach. The process starts with sequential process aiming to acquire requirements and partitioning the system development into executable components which are developed using cyclic approach one at a time based on their priority.

Another methodology described in the thesis is *A Two Layer Approach for Ubiquitous Web Application Development* proposed by Dieter Blomne et al. presented in IEEE in 2009 [5]. In this approach, customized and adaptive user interfaces are built for various web applications considering the device diversity and anytime/anywhere quality of internet access.

The last in the list of development methodologies presented in the thesis work is *Designing and Implementation of Agile Framework based on Lift* presented by HU Dong-Hong in 2010 at IEEE [16]. Lift is framework for development of web application which is based on scala programming language [45] combining advantages of popular frameworks like rails [52] and django [44]. The framework consists of pipe filter architecture in which at very first, the XML file is input to analysis engine which parse the file and generate the corresponding components. These components along with the database make input for lift framework which then outputs the information general parts.

CHAPTER 3

DEVELOPMENT METHODOLOGIES

Development Methodologies

3.1 Definition

According to Nancy L. Russo et al. *Application development methodologies are promoted as a means of improving the management and control of the application development process, structuring and simplifying the process, and standardizing the development process and product by specifying activities to be done and techniques to be used* [28].

Geoffrey Elliott defined development methodology as noun *An application development methodology is a framework that is used to structure, plan, and control the process of developing an information system- this includes the pre-definition of specific deliverables and artefacts that are created and completed by a project team to develop or maintain an application* [14]. As verb, development methodology can be defined as *The application development methodology is an approach used by organizations and project teams to apply the application development methodology framework (noun)* [42].

3.2 Importance

As Nancy defined the development methodology, these are used to control the development process so that better results could be produced in terms of quality as well as cost. A methodology is more than just a method embodying an analytical framework which is conveyed through inter-subjective representational practices and operated through a 'toolbox' of analytical methods, tools and techniques [43]. Using application development methodologies beneficiate the application to be developed in following ways:

1. *Conviction of why to do what to do* - First step in any development technique prescribed by any methodology to use advocates for feasible study of the system being planned to develop. Analysis of feasible study report may lead to deviation in objectives of the system being developed for better good.
2. *Complete and clear set of requirements* - Many techniques for requirement gathering and analysis have been described in various development techniques prescribed by chosen development methodology, using those, helps in getting clear idea of what to do and what is needed for developing the application. If one starts coding the application without having the clear objectives, development of application may lead to unwanted and unreliable product and thus leading to failure of organization.
3. *A proper guideline of how to do* - Now we have well known documented requirements set for starting development of application. But next thing is that how to formalise development

in proper steps and manner so that minimal risk would be associated with optimal benefit. Methodologies provide a way for this according to what we want to develop and what will be its running environment.

4. *Optimal formation of teams and distribution of work* - Randomly assigning the work to random people cannot lead to optimal use of workers according to their strengths. It is must to have maximum use of man power in order to get efficient and rapid product output.
5. *Minimal risk with optimal benefit* - Methodologies do not only provide ways for efficient development of project but also guides after the development in terms of deployment and maintenance plans.

3.3 Software Application Development Methodologies vs. Web Application Development Methodologies

General approach in any software application development starts with requirement gathering and specification followed by risk analysis and design and coding, and finally deployment and maintenance. What differs in requirement phase of both the techniques is due to diversity of users in web application development. In software application developers have customer to interact with and getting feedback about the requirements prepared and prioritized and also requirements are confined to some specific purpose. But in case of web application there is a problem of large set of users and also no straight way to get feedback from the same. Another problem is rapid change of requirements which is generally not the case with software application.

Another difference in development of both the applications is that web applications need to be updated frequently and thus costs much more than software applications in those, release of versions are very rare. There is a need to make some special arrangements in case of web application development to deal with regular release of updated application.

Next issue is security, while software applications have to face desktop viruses, web applications have to deal with hackers who may harm the web application adversely.

Another difference between developments of both applications lies in their documentation part. Web applications development need not be documented like software applications as quick development is demand, also rapidly changing requirement makes it infeasible to document each time before making changes due to short time span.

It is not always necessary to develop the web application completely before launching it. Rapid introduction of changes makes application better with time. Thus, maintenance is of more concern in compare to main development phase.

CHAPTER 4

APPROACHES FOR WEB APPLICATION DEVELOPMENT

Approaches for Web Application Development

This section is dedicated to a brief description of various available development approaches for web applications having their own advantages and lacking over each other. Table 2 enlists the approaches described in this section along with their advantages and disadvantages.

Serial Number	Web Application Development Approach	Concept	Advantage	Disadvantage
1	Agile Development Approach	Follows *Agile Manifesto* defining 17 key practices to be followed during web application development.	Main focus is on rapid development without having need of large documentation.	Heavy and complex web applications require a proper documentation and highly managed development teams for providing effective maintenance.
2	Object Oriented Development Approach	Every real world thing is treated as object and the objects are assigned some functions with them.	Highly flexible application development. Addition/deletion of new functionality is too much easier.	Requires expertise and a highly controlled development process.
3	UML Based Development Approach	Engineering drawing depicting requirement and functionality of the system leads the development process.	Highly managed development process with very clear presentation of complex processes.	A lengthy procedure not fitting for quick development which is most of the times essential in case of web applications.
4	Sequential Development Approach	Some steps are to be followed with no back track option.	Very first and the most basic approach fulfilling all basic development requirements.	Suffers from a large number of deficiencies like no risk involvement, no backtracking, and no option for handling rapidly changing requirements which is a very general case of web applications.

5	Reverse Engineering Based Development Approach	Already existing development model are analysed and new models are developed as per requirements.	Highly extensive exploitation of reusability concept accepting advantages of already existing techniques with removal of deficient practices.	Success rate of newly generated development model depends on quality of chosen model.
6	Knowledge Based Development Approach	Encode software knowledge into reusable components that can be accessed by non-expert software developer.	Non-expert software developer can be proved handy when there is a lack of experts.	Non-expert can never be 100% efficient replacement of an expert and requires highly attentive work to encode expert knowledge into reusable components.

Table 2 Web Application Development Approaches

4.1 Agile Development Approach

Agile software development is not a single approach but a framework which contains a family of processes which is based on some core principles, also called as *Agile Manifesto* drafted by 17 [23] well known developers in 2001 defining the agile development. There are 12 principles [34] which are followed in agile development:

1. Satisfying the customer through early and continuous delivery of valuable software.
2. Changing requirements are always welcome even late in the development.
3. Delivering working software frequently with a preference to the shorter timescale.
4. Joint working of business people and developers.
5. Trusting motivated individuals to get the job done and giving them environment and support what they need to develop.
6. Face-to-face conversations.
7. Working software is the primary measure of the progress.
8. Promoting sustainable development so that the sponsors, developers, and users could be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity, the art of maximizing the amount of work not done, is essential.
11. Self-organizing teams.
12. At regular intervals, the team analyses its performance and way of working so far and then changes its behaviour accordingly.

A recent survey conducted by Dr. Dobb's Journal shows 41 percentage of development projects have now adopted agile methodology, and agile techniques are being used on 65 percentage of such projects [7].

Agile vs. Waterfall - Visualizing the idea of Agile Methodology

To get visual view of agile approach for development, this can be compared with waterfall methodology of software development. The most important difference between these two methodologies is that waterfall emphasizes on sequential process configuring distinct phases with checkpoints and their individual deliverables, while agile methodology supports iterative approach with output of each complete cycle giving a working code which is used to respond changing requirements. Figure 2 and Figure 3 shows working idea of both the mentioned approaches.

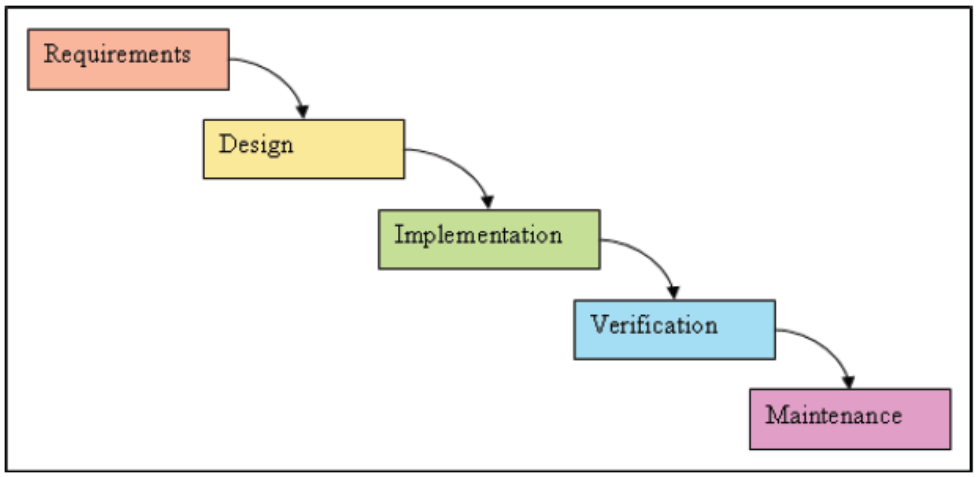

Figure 2 Waterfall Methodology [54]

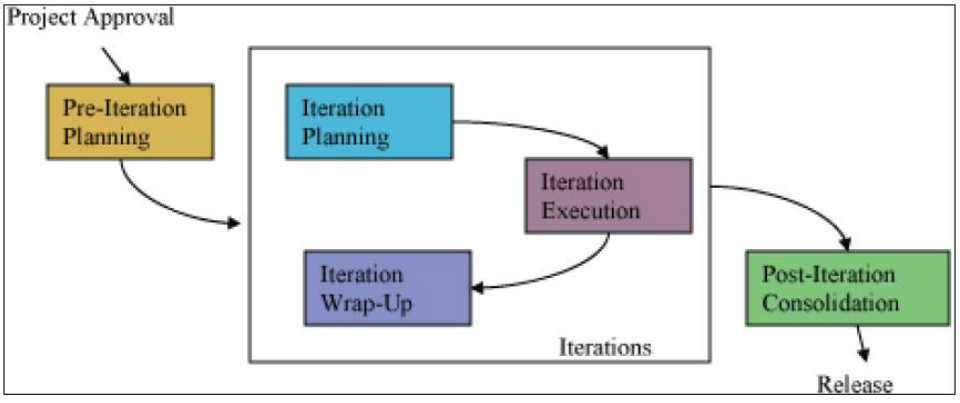

Figure 3 A Generic Agile Development Process [40]

Waterfall assumes that in very starting, it is possible to have a complete and clear knowledge of user requirements which is very rare in large scale application development resulting in deviated deliverable from what customer expected.

Agile methodologies embrace small team formation, making face-to-face communication possible, working together to define quick prototype to visualy present the problem. As shown in Figure 3, each iteration has its requirements followed by development, defining and running integrated test scripts, and finaly getting results verified by the user. Verification occurs much earlier in the development process than it would with waterfall, allowing stakeholders to fine-tune requirements while they're still relatively easy to change [40].

As specified earlier, *agile* is not a name of some development method but a term for key practices to be followed during development. Here is a listing of some of the popular development techniques [1] following agile principles:

1. Scrum
2. E**x**treme **P**rogramming (XP)
3. **A**gile **U**nified **P**rocess (AUP)
4. **T**est **D**riven **D**evelopment (TDD)
5. **F**eature **D**riven **D**evelopment (FDD)
6. **B**ehavior **D**riven **D**evelopment (BDD)
7. **Ess**ential **U**nified **P**rocess (EssUP)

4.2 Object Oriented Development Approach

Object oriented development approach emphasizes on decomposing the system into objects as components. An object is realization of real world entity and its behaviour is identified by the actions that it goes through and that it requires of other objects. Object oriented approach is different from functional approach in sense that decomposition in functional approach results in components identified by the major step in the whole process. Here is an example of cruise control system (to maintain the speed of a car) describing main principle of object oriented approach by showing difference with functional approach as shown in the Figure 4.

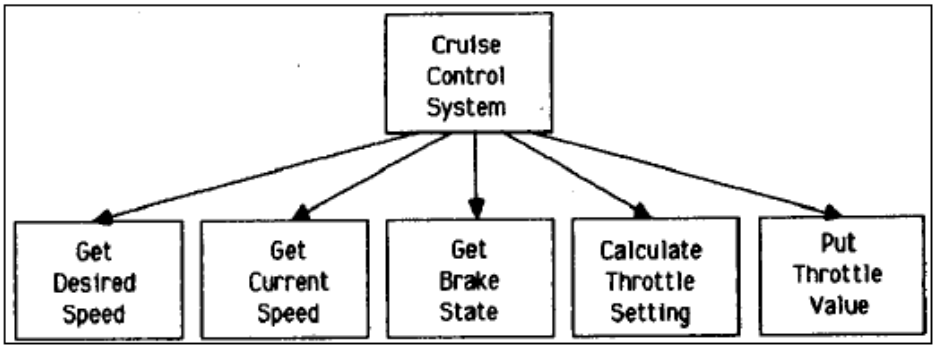

Figure 4 Functional Decomposition of Cruise Control System [13]

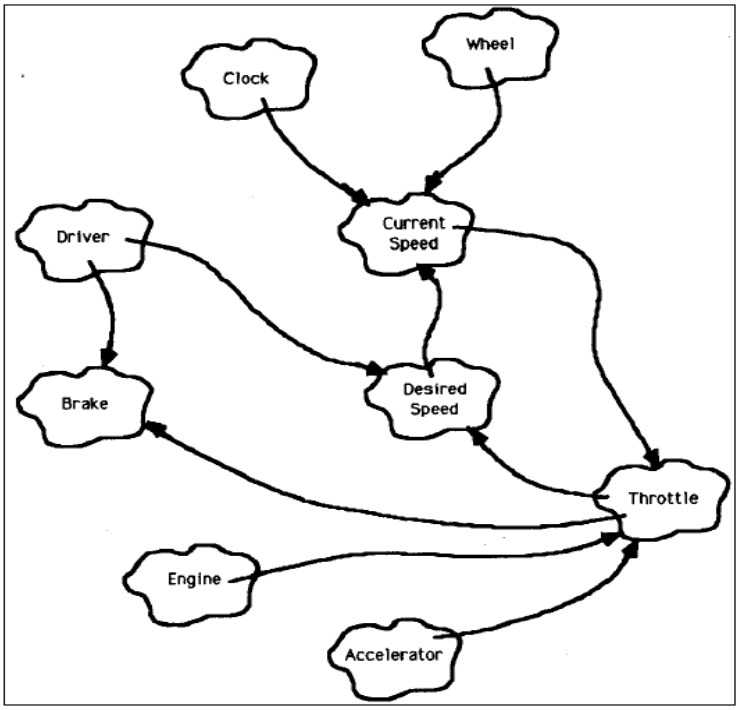

Figure 5 Object Oriented Decomposition of Cruise Control System [13]

Functional decomposition depicts components as major parts of overall process constituting the complete system when integrated while on the other hand, object oriented approach decomposes system into objects having dependencies shown by arrows in Figure 5. Benefit of having object oriented approach is abstraction. Also, handling changes in program code affects very little part unlike functional approach.

Steps of object oriented development approach evolved from an approach first proposed by Abbott are as follows [35]:

1. Identify the objects and their attributes.
2. Identify the operations assigned with each object and the ones required of other objects.
3. Establish the relationship amongst the identified objects.

4. Establish the interface of each object.
5. Implement each object.

In object oriented system, object plays the central role. Thus, it is necessary to describe properties and operations of objects. The Table 3 represents the type of operations of the objects, and following is the listing of the properties of the object:

1. Object has a state.
2. Object has a set of operations associated with it.
3. Object is an instance of a class (class is a logical view of how the objects may be formalised. A class may have more than one objects having different set of values for same variables).
4. Object is denoted by some name.
5. Visibility of each object is restricted varying object to object.

S. No.	Type of operation	Description
1	Constructor	Changes state of the object
2	Selector	Evaluates current state of the object
3	Iterator	Permits all parts of the object to be visited

Table 3 Operations of the Object

4.3 UML based development approach

The Unified Modelling Language (UML) represents various aspects of the application to be developed like design of components, communication between them, lifeline of these components etc. diagrammatically. We have various tools available for modelling the software design and patterns like *rational rose* with support of creating working code from diagrams automatically.

UML can be applied in any number of ways, but the most common uses are:

1. Capturing the requirements of the application
2. Designing the software
3. Representing communication among software or business processes
4. Documenting an existing system, process, or organization

UML has been evolving since 1990s and has its root in three distinct object-oriented methods - Object-oriented analysis and design (OOAD) given by Grady Booch, The object modelling technique (OMT) by Jim Rumbaugh, and The object oriented software engineering method (OOSE) by Ivar Jacobson. In 1996, the UML was introduces as UML 0.9 and then 0.91 leading to UML 1.0 in 1997. The current versions of the UML are as follows - UML Superstructure version 2.4.1, UML

Infrastructure version 2.4.1, OCL version 2.3.1, and UML Diagram Interchange version 1.0 [31]. Figure 6 represents the versions of UML so far.

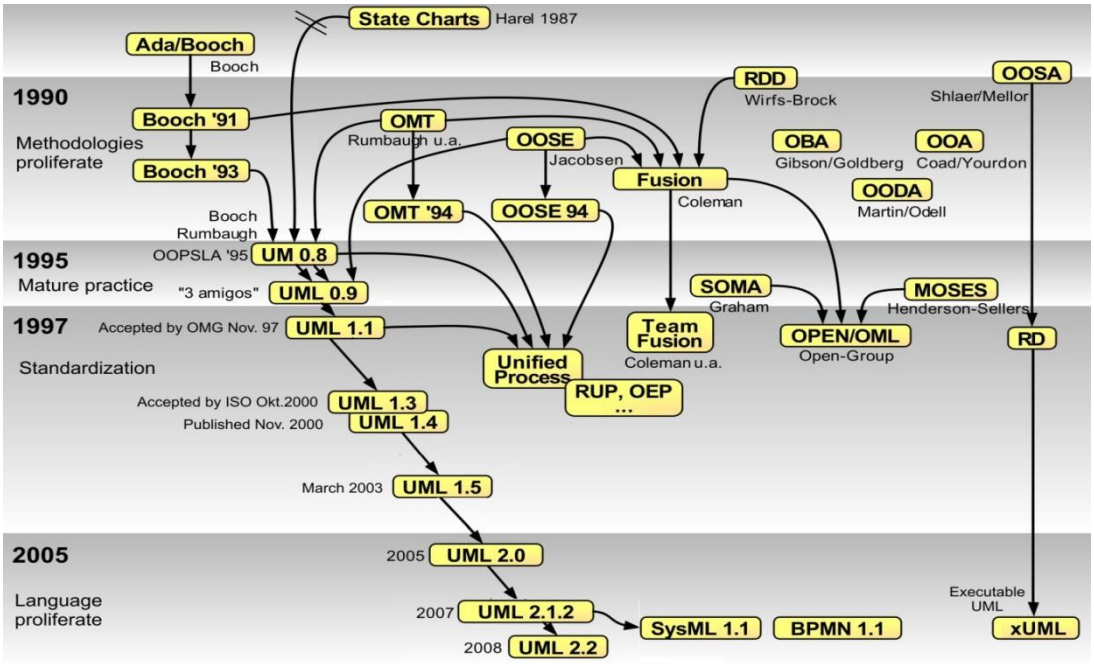

Figure 6 Evolution of UML Versions [30]

UML 2.2 has 14 types of diagram classified into two categories: Structure diagram and Behaviour diagram [48]. Structure diagrams are used extensively in visualizing the software architecture of the application and behaviour diagrams are used to visualize the functioning of the application. Figure 7 categorises the UML diagrams.

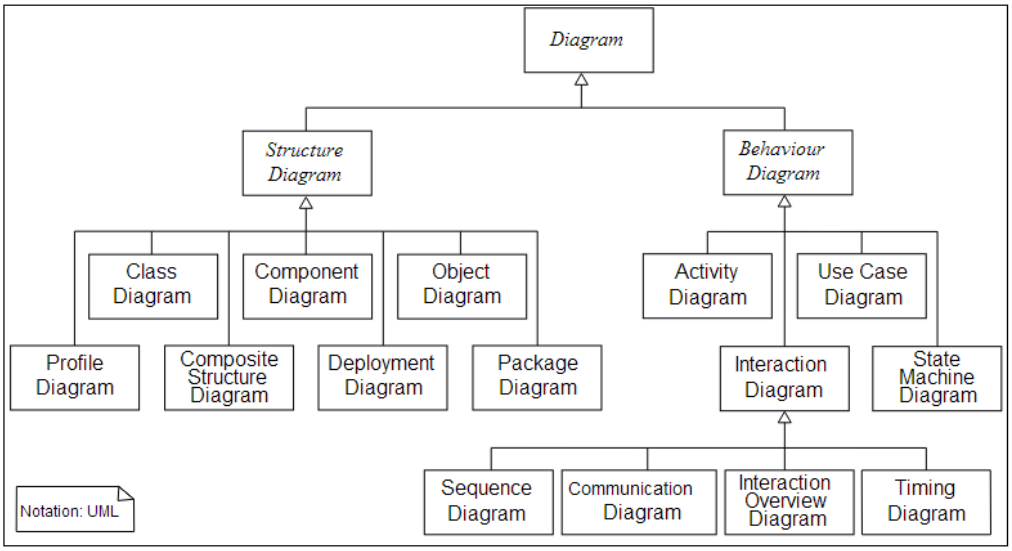

Figure 7 Classification of UML Diagrams [47]

4.3.1 Structure Diagrams

Structure diagrams are used to prepare structural view of various aspects of the application. UML includes following types of structural diagrams.

Class Diagrams – represents the attributes and functioning of the classes and relationship among them.

Component Diagram – depicts decomposition of the system into components and also dependencies among them.

Composite Structure Diagram – describes the internal structure of a class and the collaboration.

Deployment Diagram – represents the hardware used and the artifacts deployed on the hardware along with the execution environment.

Object Diagram – at a specific time, shows a complete or partial view of the structure of the system.

Package Diagram – describes composition of system into logical grouping and dependencies among them.

Profile Diagram – operates at the metamodel [18] level.

4.3.2 Behavioural Diagrams

Behavioural diagrams are used to prepare behavioural view of the web application in several manners. Following is the listing of such diagrams introduced in UML.

Activity Diagram - describes the overall flow of control showing activities to be performed.

State machine diagram - depicts the state transition of the system along with causes to make the transition.

Use case Diagram - represents the functionality of the system along with actors performing the function represented as use case and dependencies among the use cases.

Interaction Diagram - shows the flow of control and data among the things being modelled. It is further divided into subparts.

Communication Diagram - specifies the flow of messages among various objects being modelled depicting communication among them.

Interaction overview diagram - represents the overview having nodes representing communication diagrams.

Sequence Diagram - shows the communication between objects in terms of messages along with lifespan of that object in term of that particular message.

Timing Diagram - focuses on timing constraints.

4.4 Sequential Development Approach

Sequential development approach supports step-by-step development of the application with each step resulting in a product being the main constitute for the next step. Waterfall development methodology, a very general and well known approach, gives idea about sequential development approach. Figure 8 depicts the waterfall development methodology, based on which many development techniques have been evolved.

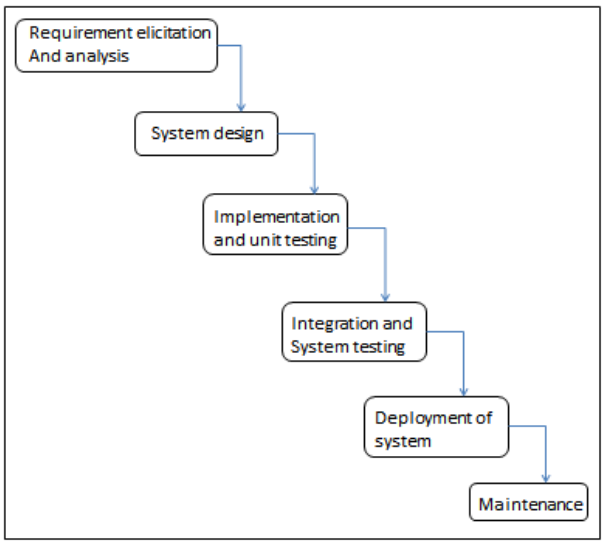

Figure 8 Waterfall Development Methodology

As shown in Figure 8, requirements to be implemented are gathered and analysed at very beginning considering it complete set of requirements and later on possibility of changes in these requirements is not considered. Output of this step is software requirement specifications (SRS) which initiates next step which is *system design.*

Next step is *system design* whose output is software design document (SDD) presenting system's architecture which is information of components of the system to be developed.

SDD is fed into next step which is *implementation and unit testing* producing code work of the components designed in the SDD as output product of the step. Also, each component developed is tested often using stubs and drivers making the component executable.

After having all the components on the desk, integration is performed in order to get complete system in a piece post successful system testing.

Next steps are launching the product and resolving customer issues based on the business plan varying corporation to corporation.

The waterfall development methodology has various flaws like high probability of risk and uncertainty, delayed working software, no loop back (stepping to previous step), compulsion of

complete and non-changeable set of requirements etc. raising some modified versions like Steve McConnell's "Modified Waterfalls" [24], Peter DeGrace's "Sashimi model" [39], "incremental waterfall model" [26] etc.

4.5 Reverse Engineering Based Development Approach

Reverse engineering is quite different approach from others in sense that it develops new application from currently existing application, though newly generated application may have exactly same functionality as of older one (implementation of existing application in different language) or this may be modified version having advantages over existing one (developing new application model having better performance and enhanced functionality) or even may be completely distinct application (entirely different functionality) exploiting pros and cons of existing application's design etc. phases. Figure 9 helps in visualizing reverse engineering approach.

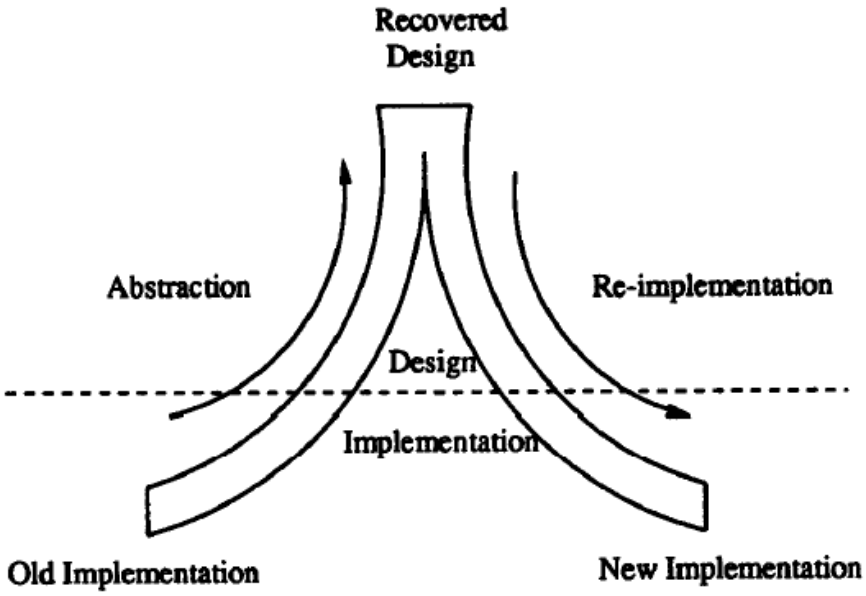

Figure 9 Reverse Engineering and Re-implementation Process [8]

There is another way to get the job done - manually rewrite the existing system, but studies declare reverse engineering approach to be more cost effective than manually rewriting the system [15][36].

Reverse engineering process starts by extracting the detailed design information from existing design documents and codes, and then it is converted into highly abstracted design documentation. The process is as shown in Figure 10 and is briefly described below.

1. *Information Collection* - All information regarding existing application is collected via all possible sources like source code, design documents, documentation for system calls and external routines etc. Also, personal experience with the application is also shared.
2. *Information Analysis* - Information collected is analyzed helping the team members getting a picture of the existing application and recording the extracted information in the desired way.
3. *Structure Extraction* - For identifying the structure of the program, set of structure chart is prepared having nodes and edges forming graph like structure. Each node in chart represents some function in the source code. Node at the connecting end is the one being called by its ancestor one and thus a hierarchy comes out as result. Also, the data entering and exiting the node is recorded for each node.
4. *Recording Functionality* - Processing done in the program routine for each node is recorded corresponding to respective node. Functionality for system routines and library routines used can be recorded in natural language or some informal notation.
5. *Recording Data-Flow* - By analyzing extracted structures, transformation of data among several routines is recorded and a set of hierarchical data flow diagrams (DFDs) is developed.
6. *Recording Control Flow* - Control flow diagrams are developed by analyzing high-level control structure of the program.

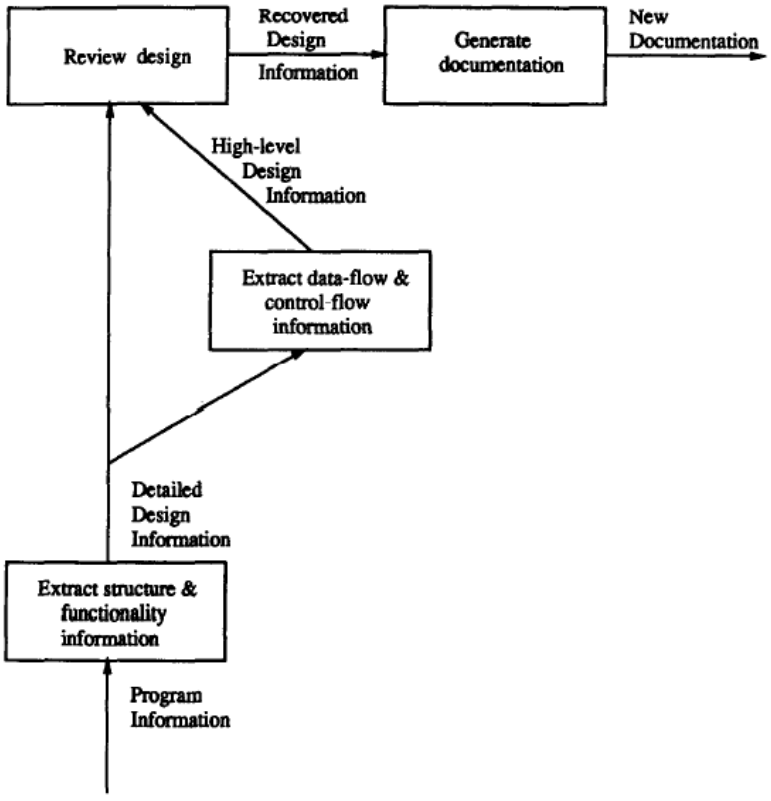

Figure 10 Reverse Engineering Procedure [8]

7. *Reviewing Recovered Design* - Recovered design is reviewed for consistency against available information and any missing item of information is identified and attempted to locate.
8. *Documentation Generation* **-** The final step, generating the design documentation, might need some more information which could not be fetched from source code like purpose of the application, system overview and history, etc.

4.6 Knowledge based Development Approach

Knowledge-based techniques facilitate the encoding of expert software knowledge into reusable components that can be accessed and assembled by non-expert software developers into programs and systems [25]. The knowledge based development approach recognizes two entities - "The specifier" (This has information about the system and it can communicate its understanding to other entity. Software engineers play this role.), and "The developer", (This holds responsible for transforming initial model to complete system. A knowledge based system plays this role). Process begins with work of specifier i.e., software engineers. Software engineers specify the requirements of the application in a high level and domain oriented form, which with the assistance of a domain oriented knowledge base of design /code component, are understood by the "developer". Knowledge base system uses the initial requirement specifications presented in form of inputs and outputs and searches for appropriate design model in knowledge base of design/code.

There are a number of artificial methods that can fulfil the role of the "developer" effectively. Mehdi T. Harandi has listed some of the appropriate artificial methods in his research paper "Building a Knowledge-Based Software Development Environment" published in1988 at IEEE [25].

1. *Knowledge Based Techniques* - Design/coding knowledge is encoded into such a form which could be accessed and incorporated in to some new software system. Knowledge base assist in synthesizing high level abstracted design, transforming it into detailed design and coding, and implementing it.
2. *Constraint Propagation Techniques* - Design constraints specified by users automatically propagate to affected design components and thus desired consequences of the specific design decision gets inferred. Likewise, implementation constraints propagate to corresponding part of the program and desired results are inferred.
3. *Planning Techniques* - This technique uses combination of knowledge structures for finding appropriate design/code model. This is required when user given specifications cannot be accounted for a single design/code model.

4. *Agenda Driven Control Strategies* - These strategies are used to control incomplete or not reflected user specifications by setting up goals to accomplish design/code decisions for such type of cases. Also, design/code conflicts are handled by scheduling goals for same.

CHAPTER 5

VARIOUS METHODOLOGIES FOR DEVELOPING WEB APPLICATIONS

Various Methodologies for Developing Web Applications

Previous section described some of the existing approaches for developing the web applications. This section provides details of some methodologies based on the approaches described earilier.

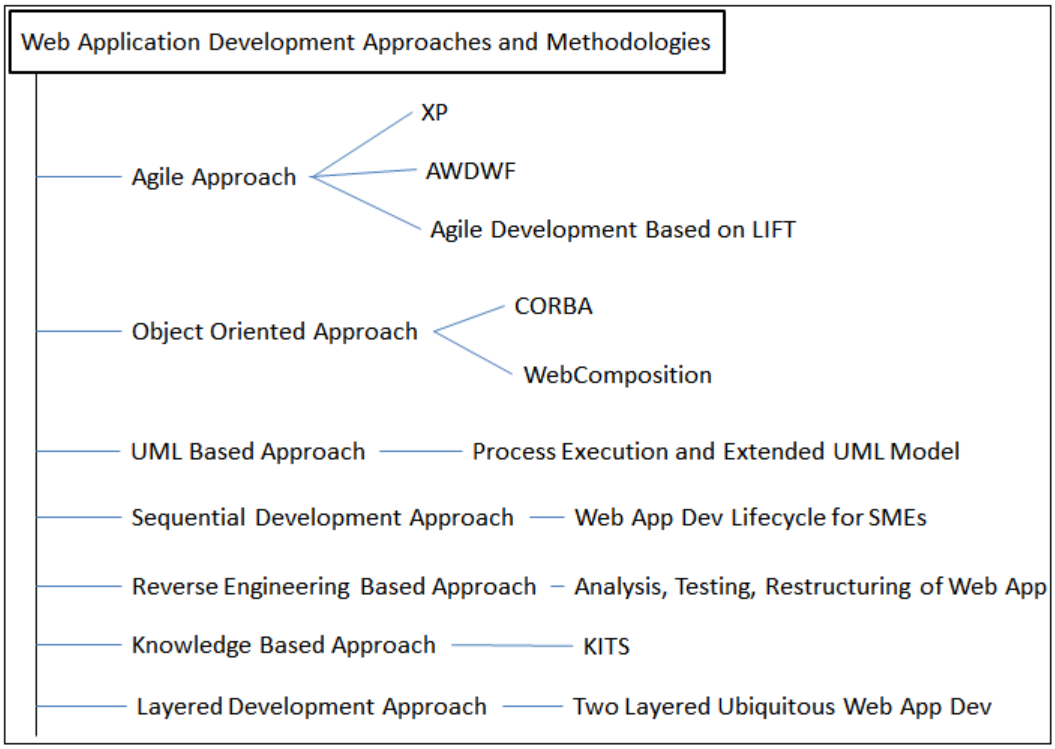

Figure 11 Web Application Development Approaches and Methodologies

5.1 Agile Web Application Development Methodologies

5.1.1 Extreme Programming

Web-based applications require faster time-to-market matric and rapid introduction of the new requirements. These requirements demand agile software processes like XP, which enhance development productivity while maintaining software quality and flexibility.

Extreme Programming (XP) approach focuses on building small teams of 5 to 15 members and follows 12 related practices. Also, XP concentrates on delivering executable code and automated test drivers rather than spending effort on paper based requirement and design documentation part. Building small teams for development helps in avoiding documentation part as face to face communication can take place effectively. But, as the development organization grows time spent exchanging product knowledge and training new people increases and often renders XP unsuitable [11].

XP proposes a set of software development practices to increase productivity while maintaining quality. F. Maurer and S. Martel grouped XP's 12 practices according to three key areas: customer satisfaction, software quality, and development process organization [11].

Key Areas	Practices	Remarks
Customer Satisfaction	On site customer	Works in parallel to development team in order to clarify requirements and verify end results.
	Small releases	Rapidly changing requirements are handled by releasing end products in short spans.
Software Quality	Metaphor	High level software architecture.
	Testing	Unit tests are written before writing actual code.
	Simple design	Helps in handling issue of minimal documentation.
	Refactoring	The code is refactored until there is simplest way to run all the tests on developed feature.
	Pair programming	Two persons work simultaneously on the same machine and switch their role after completing their task.
Project Management	The planning game	Scope of next release is decided.
	Sustainable development	Product quality is preferred over high turnover by avoiding overtime work and burned-out situation of developers.
	Collective ownership	Developer owns right of the part of the code developed.
	Coding standards	All work is done following coding standard in order to make collaborative working cost effective.
	Continuous integration	Emphasis is on short releases as early possible.

Table 4 Key Practices being followed in XP Web Development Approach

First key area is getting customer satisfaction which is practised using two practices - On site customer, and small releases. Web development process starts with capturing requirements and prioritizing them. In XP, requirements are documented through user-stories, which are basically textual use case description. An *on-site customer* representative is elected in order to clarify the requirements and also setting the priorities and he works with the development team in parallel. As web requirements change very rapidly, *short releases* are made continuously which ensures that each released product produces significant business value for the customer.

Next key area is keeping application quality high while doing some other practices. This includes five practices which might appear unusual but their combined application affects the quality issue by great significance. A very first practice is *metaphor* which represents a coherent view of the system that makes sense to both the business and technical sides and represents "what we are trying to do" thus serving as the high-level software architecture [11]. Next is *testing* in which unit tests are written before writing actual code serving as a detailed specification of the method's functionality which helps software developers to think about the problem before programming. Next one is about keeping the most possible *simple software design* which improves the team's ability to work productively with minimal documentation beyond the source code [11]. When source code is easy to understand, there is no need to document its structure at a higher level of abstraction. Programmers change the code's structure to improve its understandability and maintainability, without changing its functionality [21]. Once a feature is complete, the existing structure is examined whether it is the simplest way to run all the tests or not, in that case the code is refactored and simplified, another practice for software quality - *Refactoring*. Another practice to be followed in order to maintain application quality is *pair programming* according to which all production code is written by two people working at common machine. At a time one of the persons controls the keyboard and the mouse, and keeps focusing on broad issues like simplifying the approach and finding its working possibilities etc. The other person works more strategically and decides if the way being followed to implement the functionality is the best one. The pair switches roles frequently throughout the day. In situations where pair programming is unacceptable, teams might consider replacing it with software measurements and inspections [19].

Next issue is project management which aims at reducing management overhead, while giving customer's interest highest priority and includes overall five practices - the planning game, sustainable development, collective ownership, coding standards, and continuous integration. *The planning game* is a scope deciding practice for next release of product considering both business priorities and development team realities. To make such decision business personals need some information from side of technical team. Developers provides their report based on three basis -

Estimates (effort required to fix a bug or to include some new feature), Consequences (impact of various business decisions on development process), and Process (a complete team organizing plan in order to work together). Next practice is *sustainable development* which pleads for flexible and refreshing work schedule of developers as pushing development team to work overnight cannot result in high quality product. In *collective ownership*, anyone who make some modification or add some code work, is given ownership of that part of code which makes it easier to get feedback about their part of code whenever it is tested or modified. Following *coding standard* helps developers to work in collaboration with their teammates and also makes it easier to understand and modify other person's code. Next practice is *continuous integration* emphasizing on the fact of always having an executable product available. Code is integrated as frequently as possible and thus providing an executable which is tested by a separate team dedicated for this purpose only.

5.1.2 Agile Web Development with Web Framework (AWDWF)

AWDWF exploits advantages of agile development using web framework. It is an excellent Web development technique for the Web application system which fulfils the requirement of quick service, quick response, and adoption of rapid changes [17]. AWDWF emphasize on communication between people for example, the communication among the development team members, and between the customer and development teams. The success of project depends on regular and continuous customer feedback. In order to handle rapidly changing customer requirements, prototypes are released frequently and this helps in quick and timely customer feedback. This process of receiving feedback, testing the product and then evaluation take place throughout the entire life cycle.

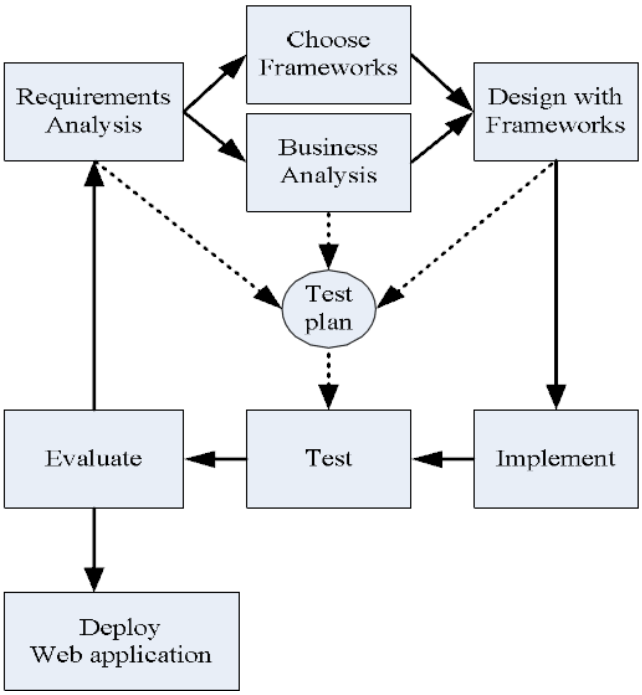

Figure 12 Main Process of AWDWF [17]

The main idea behind AWDWF is keeping development team focused on the core business logic and thus avoiding programming duplication and wastage of resources, consecutively, improving system's efficiency and stability, and maintaining quality of the web application. AWDWF is based on the web framework and therefore, the most of the web application system is based on the MVC multi-level structure to develop [17]. After requirement and business analysis web framework is chosen for every layer by analysts. As shown in the Figure 12, AWDWF uses a combined approach - *incremental and iterative*. In each iteration, the main problem causing that iteration is highlighted and then the same incremental approach is applied to solve the problem and thus giving a rapid prototype in order to collect customers feedback on the change made and same process is repeated if some other problem is found.

In AWDWF, there are three main personnel among whom communication takes place - user, web developer, and framework developer. Figure 13 shows the type of communication between above specified entities.

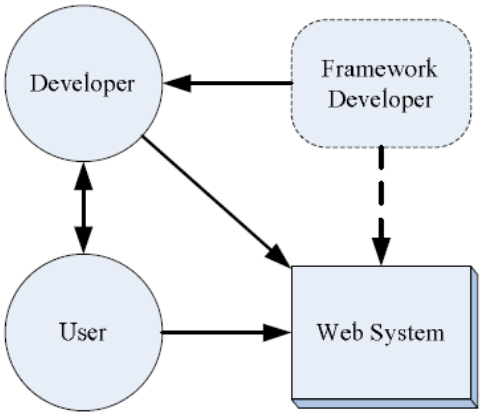

Figure 13 AWDWF Team-to-Team Communication [17]

Framework developers communicate with web application developer in order to make web framework for the iteration. Also, being third party, they analyse the web application being developed using the framework chosen by them and if necessary, modifies or change the framework for next iteration.

Between developers and users, a bi-directional communication exist which shows continuous interaction between both of those which is because of feedback being received from the user side after demonstrating them quick prototypes.

5.1.3 Agile Web Application Development Based on Lift Framework

Lift is framework for development of web application which is based on Scala programming language [45] combining advantages of popular frameworks like rails [52] and django [44].

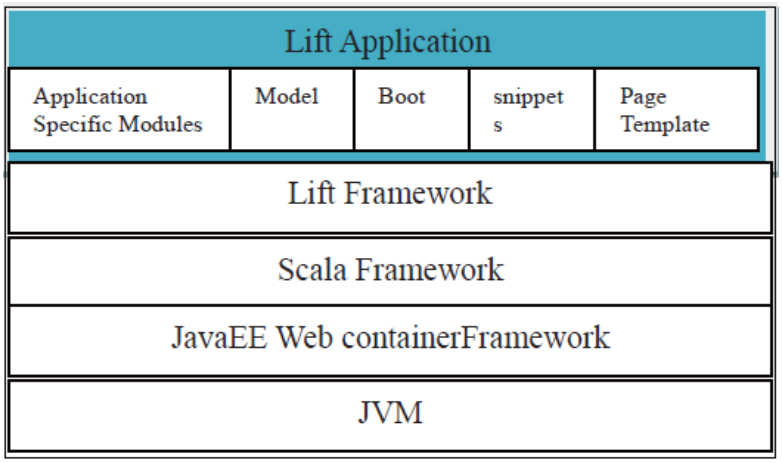

Figure 14 Lift Framework [16]

In Figure 14, the very first layer represents the agile application framework which is here point of interest and HU Dong, and Yu Xue have described this framework in their research paper "Designing and implantation of agile framework based on Lift" published in 2010 at IEEE. Figure 15 depicts the agile framework's position.

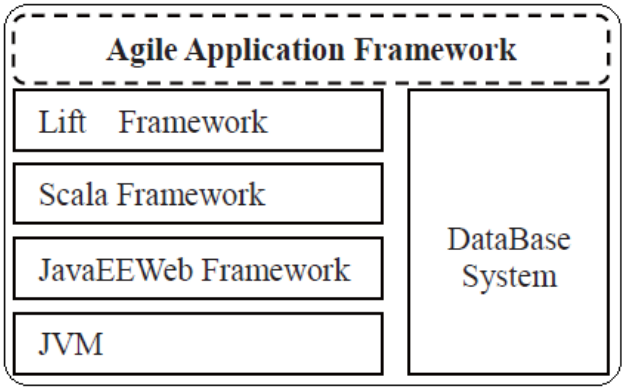

Figure 15 Agile Framework's Position [16]

5.1.3.1 Designing of Agile Application Framework

Figure 16 shows a pipe filter architecture depicting main components and work process of agile framework. The XML file is input to analysis engine which parse the file and generate the corresponding components. These components along with the database make input for lift framework which then outputs the information general parts.

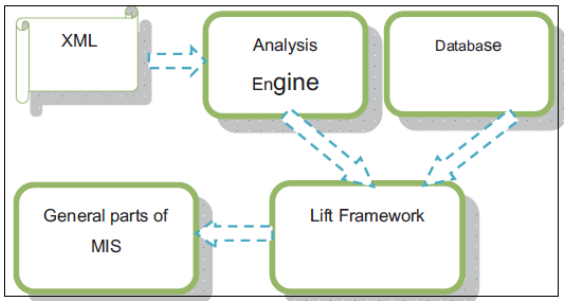

Figure 16 Pipe Filter Architecture [16]

The Figure 17 shows architecture of analysis engine, one of the kernel components, which parses the input XML file and generate page dictionary and produces various components like DB components, template components, and menu components which later on produces modules.

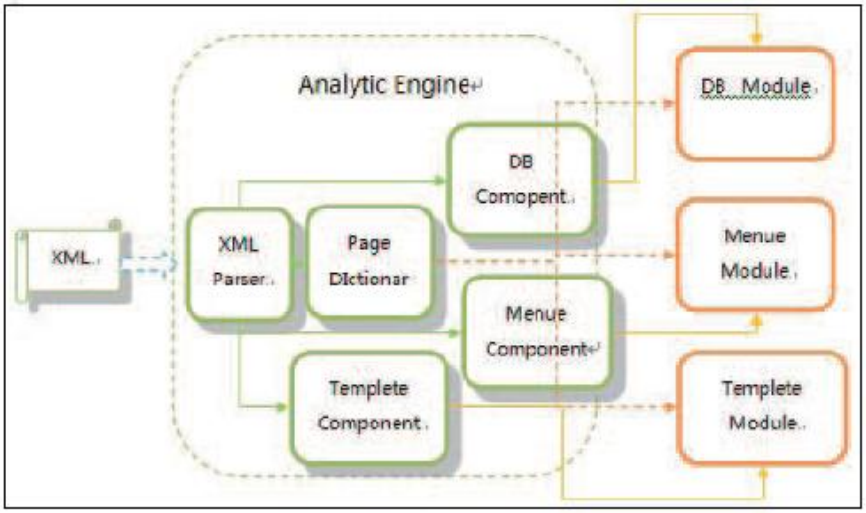

Figure 17 Analysis Engine Architecture [16]

5.1.3.2 Implementation of the framework

A. *Analysis Engine Subsystem* - XML analysis engine is based on javax.xml which is a standard java library and its main function is transforming the XML information into Document Object Model (DOM) tree structure and then analyzing them using Xpath mechanism [57].

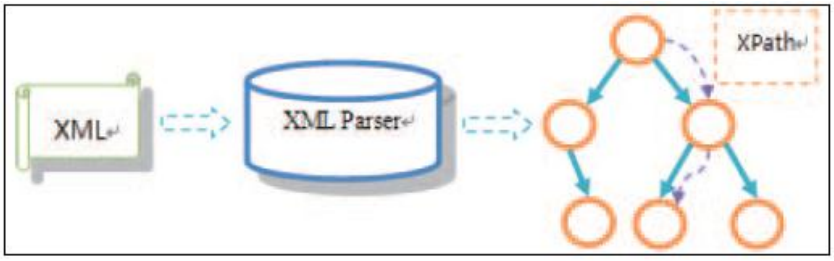

Figure 18 Relationship of XML Engine Xpath [16]

B. *Menu Generation Subsystem* - Page information is converted into the main menu of the system. Also, menu structure, page view style, and access control strategies are defined by menu generation subsystem.

C. *Template Generation Subsystem* - This is one of the kernel function of the framework responsible for generating different components like add, delete, update, query etc. Figure 19 shows the process of template generation.

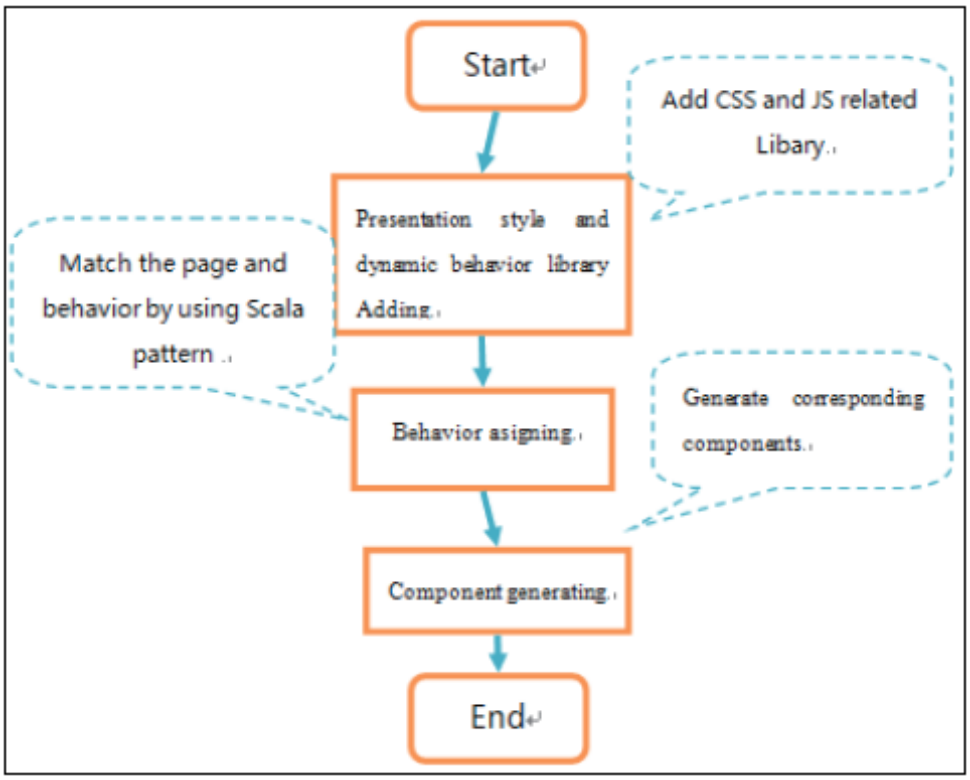

Figure 19 Template Generation Process [16]

The process is described by following points:

1. *Presentation style and dynamic behavior library adding* – Cascading style sheets (CSS) and related javascript library are used for generating and adding component presentation style.
2. *Behavior assigning* – Components are assigned pages' behavior by their behavior classification. The pages are classified by functions - view page, dumb (blank) view page, error page, data adding page, and data management page (view with action).
3. *Component generation* – Corresponding components are generated.

5.2 Object Oriented Web Application Development Methodologies

5.2.1 Web Application Development based on CORBA

Common Object Request Broker Architecture (CORBA) is an architecture that is proposed by OMG [29] for object oriented distributed computing environment [41]. CORBA provides service suit which includes services like naming, events, trading, security, externalization, licensing, concurrency, transactions, persistence, time, and so on [37]. It allows different implementation languages in server and client.

In CORBA, objects are accessed using Object Request Broker (ORB) which maintains two databases: Interface Repository, and Implementation Repository. CORBA based architecture is

shown in the Figure 20, where ‘C’ refers to an object playing the role of a client, and ‘S’ refers to an object playing the role a server.

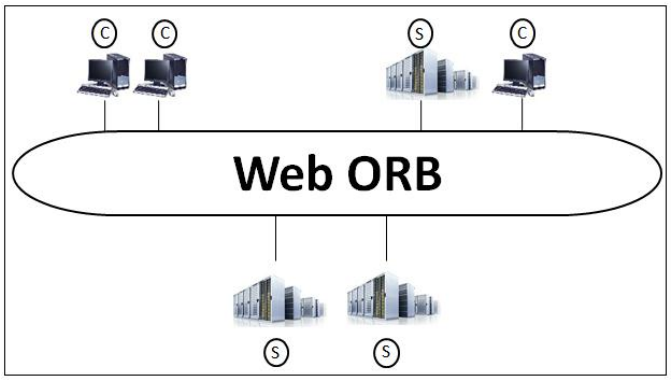

Figure 20 CORBA- Based Architecture

A site with only ‘C’ objects is regarded as traditional client host; a site with ‘S’ objects only is regarded as server; some may include both ‘C’, and ‘S’. Thus this model is effective for establishing peer-to-peer distributed systems. CORBA architecture provides fast web application through load balancing. Client can access all of the objects without physical location because both process and data distribution are supported by CORBA. Following points explains CORBA in brief -

1. *Object partitioning* - There may be three types of object partitioning - client objects, server objects, and client/server objects. Main advantage here is that almost all objects may play role of client and server because each object can access the method of other object without physical location.
2. *Process/Data distribution* - It means that local machine at client side should process the data thus reducing the network traffic and enhancing response time.
3. *Multiple services* - CORBA provides various services as named earlier which supports the integration and interoperation of the distributed objects. For example, Object Life Cycle service defines how CORBA objects are created, deleted, moved, and copied.
4. *Synchronization* - For efficient client/server system, blocking and non-blocking [6], two model are considered. In non-blocking model, clients can process other tasks after issuing request.
5. *Management of services* - ORB is a dedicated control mechanism that mediates interactions between client applications needing services and server application capable of providing those [9].

6. *Linking legacy system* - To make it easy to settle communication between various network resources, object wrappers (object oriented interfaces to legacy code) may be applied to the resources.

5.2.2 WebComposition (Object Oriented Web Application Development)

In 1999, W. Gellersen, and Martin Gaedke proposed an approach 'WebComposition' for developing web application using object oriented principles. 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 [50]. Figure 21 shows the main idea of WebComposition which is composed of mainly two parts - component model, and Markup Language. A resource generator maps the component model to a standard web implementation. The overall process is deriving Web view from developer's view of an application which is maintained by component model.

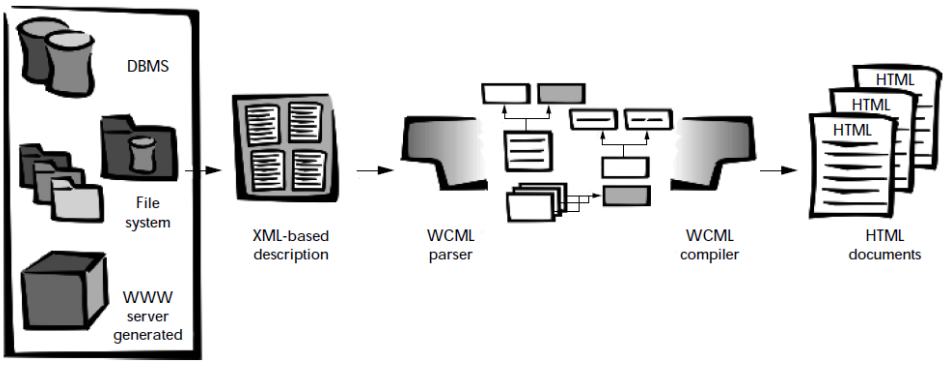

Figure 21 WebComposition Architecture [50]

5.2.2.1 Component Model

This is an object oriented model which models web entities at arbitrary level of abstraction and granularity exploiting components as a uniform concept. Design artefacts are captured by components at their natural granularity. For instance, components can capture a content unit as design artefact independently of a Web page, which is a separate design artefact [50]. Here arbitrary granularity of component means, web entities like individual links or web entities like a complete HTML page, both can be modelled as a component.

Component model also facilitate the concept of aggregation, and specialization, for instance, some component which is modelling a header part of the HTML page can be referenced by some component which would be modelling some other HTML page. For another example, a component

which is modelling the navigation structure of the web application can reference another component which might be modelling all the links and anchors involved in that navigation structure.

By means of a special reference type, components can reference so-called “prototype components” from which they inherit state and behaviour, also any component can function as a prototype [50]. 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 [4].

In order to implement the web application from the component model, it is essential that each component implement a service to map its state actual representation in web page. Though, If HTML page is modelled as a component then this mapping does not apply on the page because, for instance, HTML can be mapped to technologies such as CSS and XSL. Further, to implement this mapping, the resource generator comes in place and performs the mapping of component into “Markup Language specification”, the other main part of WebComposition.

5.2.2.2 WebComposition Markup Language

WCML is based on XML language which is a meta language facilitating user defined tag based textual format. The figure – 21 shows an example of structure of a WCML document.

```
<wcml>
<component id='CHeader'>
   <property name='text' value=''/>
   <property name='level' value=''/>
   <property name='content'>
      <H<<refprop name='level'/>>
      <refprop name='text'>
      </H<<refprop name='level'/>>
   </property>
</component>
<component id='CFooHeader'>
   <prototype is='CHeader'/>
   <property name='text' value='This is a level 2 header'/>
   <property name='level' value='2'/>
</component>
</wcml>
```

Figure 22 Code describing the structure of a WCML document [50]

The XML document type definition of WCML describes a markup notation for WebComposition concepts i.e., for component descriptions, properties, and relationships.

5.3 UML Based Web Application Development Methodologies

5.3.1 Web Application Development by Supporting Process Execution and Extended UML Model

Although UML is one of the commonly used modelling language for modelling software projects but when it comes to web application development, it is not sufficient. Wookjin lee et al. [55] proposed an agile web development technique which consists of Navigation Model (NM) and Component Communication Model (CCM), extended from state machine package and interaction package of UML 2.0 [49], respectively. This technique uses two more features of UML i.e., conceptual model, and architecture, drawn using class diagram and component diagram without extension, respectively.

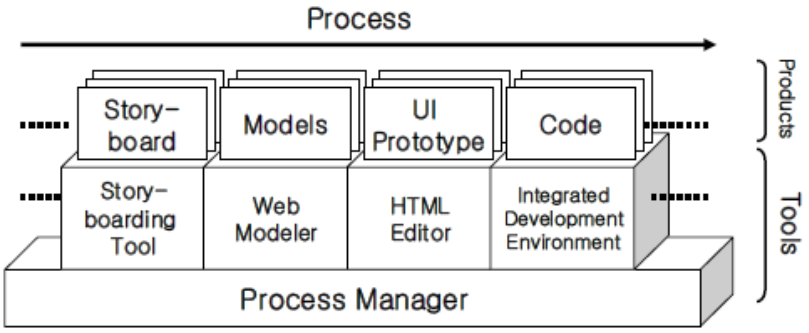

Figure 23 Overview of Methodology [55]

Figure 24 shows the agile development process, described through the role of four entities, software analyst, UI developer, component developer, and client, participating simultaneously.

First work product shown in the figure is storyboard which is representation of part of web application developed in the current active cycle along with its interaction with user. Story boarding is performed by software analyst who defines the goals of the web application.

By using the information provided by the storyboard, architecture and user interface prototype are produced. UML component diagram is used for designing architecture. For confirming the UI design, client gives his report formalized as attribute wise such as layout, images etc.

The component developer implements the components, which constitute a business logic layer of the currently developed Web application [56]. Finally, the components created by component developer, UI prototype confirmed by clients and DB system are combined into a subsystem of Web application. Each development cycle produces a subsystem and their integration after performing test activities gives final web application. This whole process is charted into Figure 24.

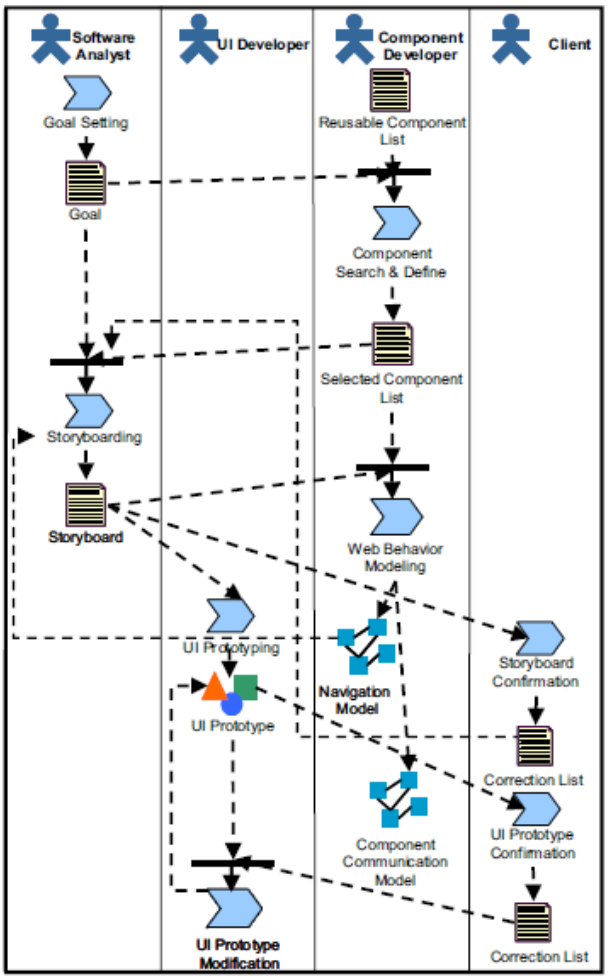

Figure 24 Web Development Process [56]

As discussed above, here we have four actors - software analyst, UI developer, component developer, and client. Software analyst analyses the requirement documentation and sets the goal which is represented in form of stories and forwarded for UI prototyping, Web behavioural modelling, and confirmation for cross checking against actual requirement set. Component developer develops the component and also selects already existing components for the purpose of reuse. Finally, navigation model, and component communication model are prepared by component developer and UI prototype developed by UI developer is sent for confirmation to client, what after correction list is sent back to UI developer and we get modified UI prototype.

Author extended AgroUML [55] tool to support web behaviour modelling by adding his model elements and diagrams by extending elements of UML 1.3 (AgroUML supports UML 1.3).

5.4 Sequential Development Methodologies for Web Applications

5.4.1 Web Application Development Lifecycle for Small Medium-sized Enterprises (SMEs)

Wei Huang et al. [53] proposed web application development lifecycle for SMEs which published in IEEE in 2008. The approach is shown in the Figure 25, which looks like combination of classical water fall model and spiral model of software development. Waterfall model is a linear and sequential model developed for software development but it does not fit in the rapidly changing requirement scenario. On the other hand, spiral model could be difficult to control without proper project management support though it incorporates the risk analysis. Wei Huang et al. developed a lifecycle for development of web application which is both sequential and iterative exploiting features of both type of approaches.

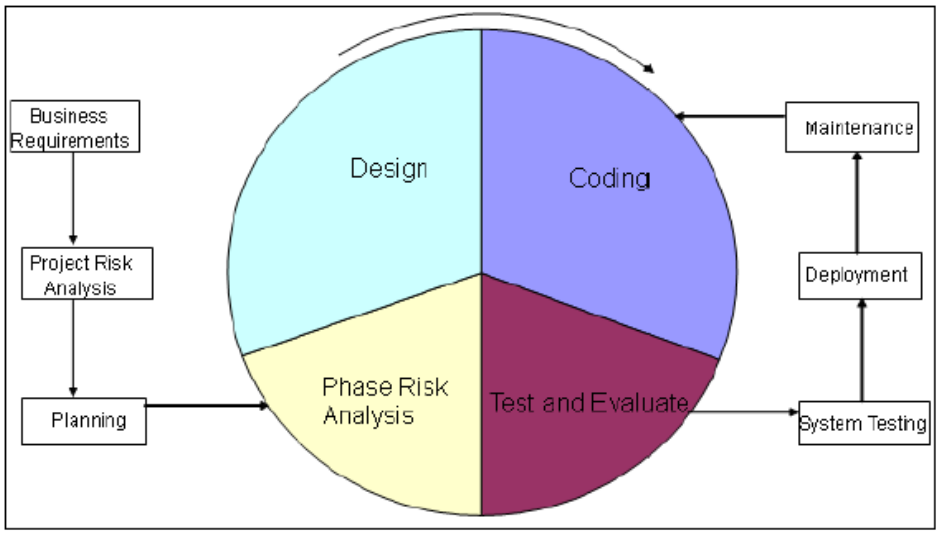

Figure 25 Web Application Development Lifecycle for SMEs [53]

5.4.1.1 Sequential process - Sequential process is aimed to perform preliminary tasks for executing iterative process by providing it the whole plan of application development. Sequential process includes following steps -

1. *Business Requirements* - Objectives of application to be developed are defined, and analyzed for feasibility. Requirements are elicited in starting phase but they can be modified or added at later time in design phase thus handling rapidly changing requirement, very general problem in developing web application which make it different from traditional desktop application development process.
2. *Project Risk Analysis* - web application development is full of uncertainty and a large number of high level risks, such as misunderstood requirements, unrealistic schedule, rapidly

changing requirements etc. This stage produces documentation defining all the possible risks associated, and their prioritization.

3. *Planning* - This phase focuses on identifying components in abstract form, relationships, and functions. A matching list of functions against resources assigned to them is prepared, based on which, a list of phases and their priorities are defined. Each phase represents a working module of the system.

5.4.1.2 Iterative Development Process - Iterative development process handles actual development part of the application and also changing requirements. Each phase identified in the *Sequential Process* phase is fed into *iterative development process* phase based on their priorities, thus developing running module of the web application.

Specific risk analysis for each phase makes the code reusable, easily scalable, easy to maintain, and robust. Based on the result of risk analysis report, a detailed design for each function is prepared including structure design, interface design, navigation design, data structure design, components design etc.

After completion of design phase, implementation of design takes place and executable codes are set for testing. For testing purpose, a test document for functional and non-functional requirements is produced, and each phase is tested and assessed against this testing documentation.

5.4.1.3 Iterative Release Process - After completion of development process of each phase, system testing is performed on the integrated complete system by people not involved in development process. The focus of system testing is to have destructive attitude and testing behaviour of application and the believed expectations of the customers.

After successful completion of system testing, the application is deployed on the server and made available for use of users. After deploying the application successfully, due to rapidly changing requirements, there is a frequent need to modify the application. This modification does not propagate back into sequential development due to limited time and management control, but it is necessary to go through the light weight iterative process for any risk identification associated with change.

A typical Web-based applications development project in an SME usually has a small budget, few resources, and tight deadline, and is bound to continuously evolve and change to meet the changing/growing requirements. This methodology fits very suitable for these requirements.

5.5 Reverse Engineering based Development Methodologies of Web Application

5.5.1 Analysis, Testing, and Restructuring of Web Applications (Based on Reverse Engineering)

In 2004, Filippo Ricca presented a research paper "Analysis, testing, and restructuring of web applications [10]" proposing an approach to develop web application based on reverse engineering paradigm. The following figure outlines the approach of creating new models for web application development from existing models having additional features and to verify better quality of newly generated model, F. Ricca developed a *prototype toolkit* [10] supporting developers in testing and restructuring activities.

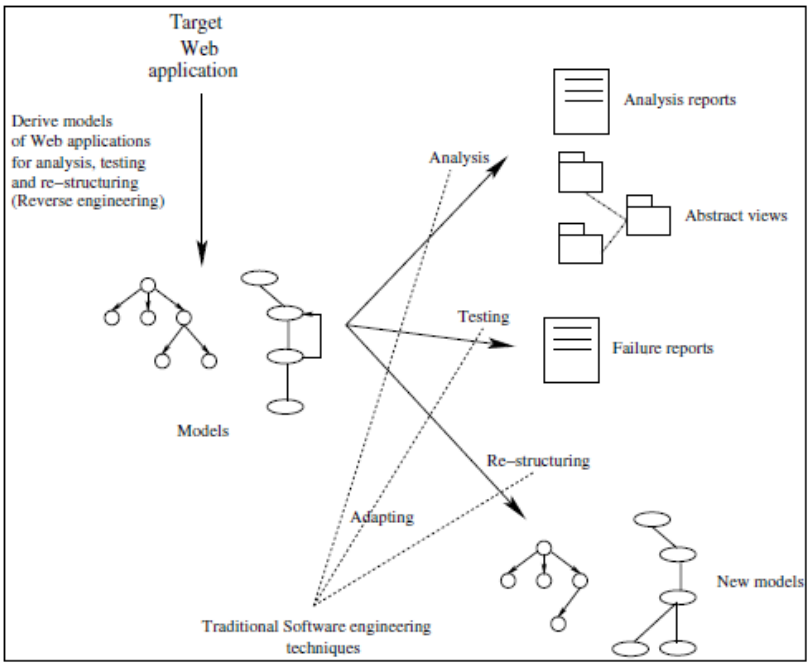

Figure 26 Reverse Engineering Approach to Develop New Web Application Models [10]

As shown in the Figure 26, after getting models of web applications, analysis, testing, and re-structuring is performed in order to get new models. Analysis is performed to better understand the internal structuring of the web applications which helps in maintenance activities and measuring quality matrices of that application, and also this helps in re-structuring step.

After analysing analysis reports, testing is performed in order to improve the quality and to detect the failures causing deviation of the application from the expected behaviour.

Finally, re-structuring is performed, which is transformation in internal structure, improving the quality of web application without changing external behaviour, and/or getting the desirable results, and thus developing new web application by transforming older one.

5.6 Knowledge Based Web Application Development Methodologies

5.6.1 Knowledge Based Integrated Questionnaire Software (KITS)

KITS (Knowledge based Integrated quesTionnaire Software) was proposed by Makoto Yoshida, and Noriyuki Iwane in the research paper "Knowledge-based Experimental Development of Web Application Systems" published at IEEE in 2005. KITS consists of three sub-system - the questionnaire subsystem, the software product line subsystem and the knowledge database subsystem [22]. Knowledge database stores metadata which resides in the form of function/data tables. The information, knowledge base stores, is domain specific information and questionnaire along with system information. The elements in the trees can be linked by the semantic relations like "same_as" relation, and this makes the independent defined elements as a unified tree and makes possible to access the related information through XML [33].

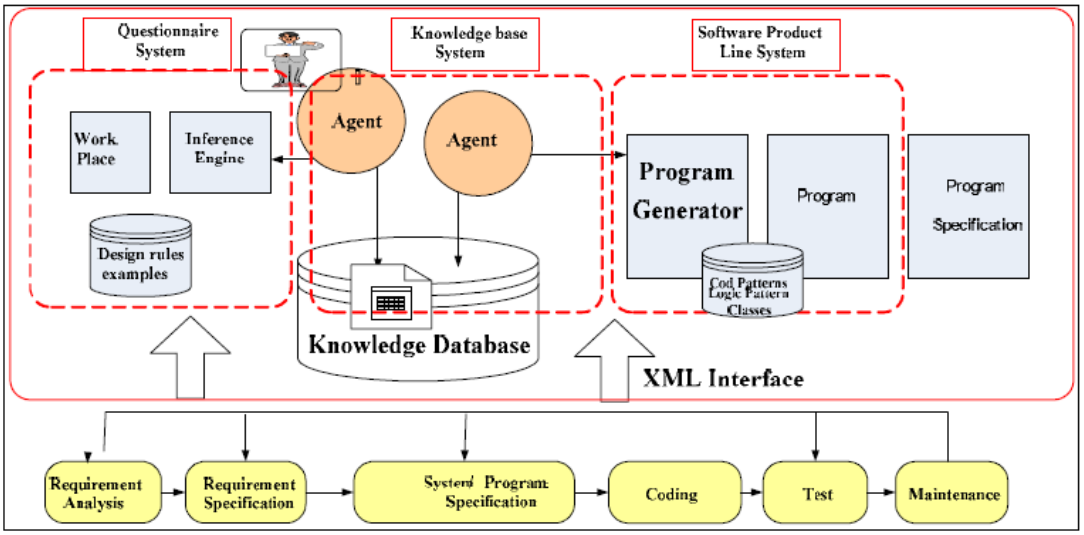

Figure 27 KITS Model [22]

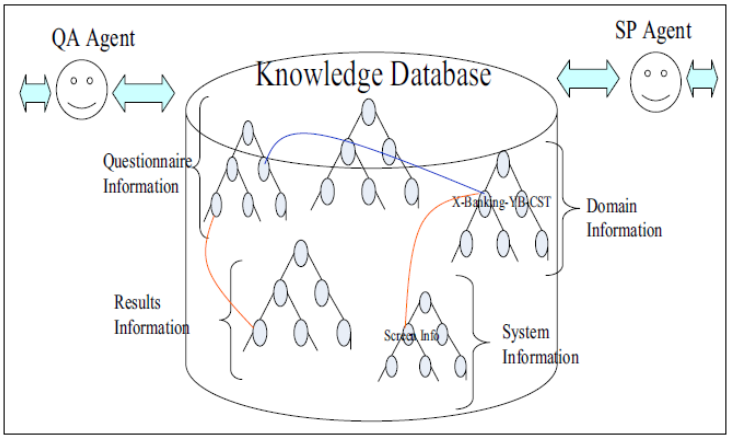

Figure 28 Knowledge Data Base Structure (Showing Banking Specific Knowledge Tree) [22]

User requirements can be easily captured by developer/ development manager in advance through KITS. Also, KITS makes it possible to develop components in advance even before development gets started. The more the questionnaire system reflects the user requirements, the closer the software product line system is to the customers [33]. Main advantage of KITS is having automatic generated code whose efficiency and completeness depends on knowledge bases' knowledge tree.

5.7 Web Application's Layered Development Methodologies

5.7.1 Two layer approach for ubiquitous web application development

Dieter Blomme et al. proposed a two layer approach in their paper "A Two Layer Approach for Ubiquitous Web Application Development" published in 2009 at IEEE for Building customized and adaptive user interfaces for various web applications considering the device diversity and anytime/anywhere quality of internet access.

Most of the new devices have limited user interfaces and restricted resource capabilities. Users wishes to have a kind of web application that can use all the functionalities and services available on the device like GPS (Global Positioning System), contact list etc. This can be achieved by series of steps. Initially the structure of the web page needs to be studied. The rearrangement of content of the page shall be done to provide easy access to users [58]. Next step is the *semantic and syntactic analysis* to provide semantic enhancements for a user on the presentation level [38]. Final step is designing a ubiquitous customizable web application [12] using *reflection* which is a key mechanism to adapt an application to changing context and changing requirements.

Web application consists of 3 layers as shown in Figure 29.

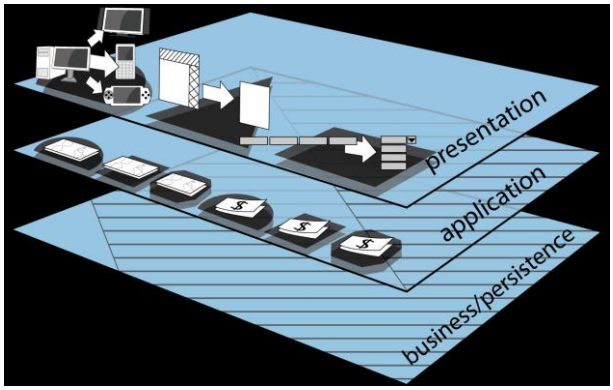

Figure 29 Three Layered Representation of Web Application [5]

Solution 1

To overcome the restrictions on user interface on different type of devices, the first location which needs to be taken into account is *presentation layer*. Here aim is to embellish the content adaptation

process by considering both the structure and content of the page and usability rules and constraints for the device.

Semantic data - while performing semi-automatic or manual process, application designer performs an intuitive sematic analysis of the page and decides what needs to be transformed. But now this process needs to be automated that takes syntax and semantics into consideration.

Techniques uses for this formal representation:

- *HTML enhanced with semantic annotations*: The HTML file itself contains the structure of the webpage. The standard used for this is *microformats*.
- *HTML 5* is used for designing web applications. It also offers new elements that are semantic replacements for common uses of generic elements.
- *RDF is based on the idea of making statements about resources*, in particular Web Resources, in the form of triples (subject-predicate-object expressions). RDF is somewhat an extended way of annotating HTML with semantic annotations.

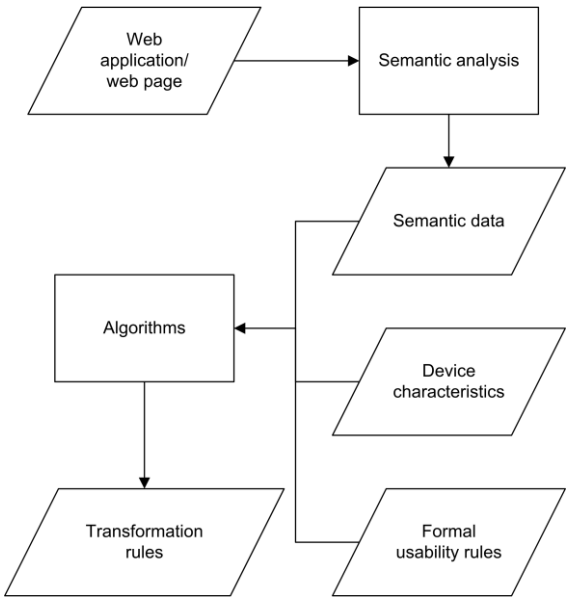

Figure 30 Workflow of Data to Generate Transformation Rules [5]

- *Formal description of usability rules*: Every device has their own type of features. The appearance of the web application should be altered according to the device that is being used to access it. This can be done by considering the usability rules widely spread on the internet.
- *Transformation rules generation*: These rules are needed to adapt the page to the particular device. These rules can be components to be used in a "*pipes and filters system"* or can be external Web Services.

Solution 2

Next target location to achieve the goal is application layer. The application logic of web applications is often split between the server and the client. To overcome the limited client-side resources and capabilities, the application logic need to be modified according to the device on which the web application is run.

This can be done by using dynamic composition [32] of interchangeable building blocks. This process can be considered as dynamic creation of web service mashup. The components can be assembled at runtime on the server.

Some implementation technologies:

- *ECMA-support:* ECMA-script uses lightweight core component that supports the most basic functionalities and that can dynamically load additional modules. By using device detection module Server-side support can be provided to detect the user agent and a javascript composer module that dynamically composes a module with the needed functions tailored to the user's device and browser.

- *RIA support:* A lot of web applications use rich internet technologies such as Adobe Flash, Microsoft Silverlight, JavaFX or AJAX. Tools like FlashProxy [27], and inspect the bytecode and transform the flash object to an AJAX-based remote view that uses server-side rendering to visualize and control the flash object.

CONCLUSION

Conclusion

Popularity of web applications has been increasing very rapidly for last some years. Web applications differ from software applications mainly in sense of their users. Also there are some other important factors too like there is no need of installation on clients side, in fact web applications are accessed via server which has to ensure better user experience on his/her host machine along with security from online threats. For these reasons, web applications development acquires different approach for development than the software applications. Also, requirements change very rapidly in case of web applications and thus resulting in more controlled and fast change management. Though, approaches being used for developing software applications are same as of web applications like object oriented, agile etc. but techniques being used for software applications cannot be applied on development of web applications due to requirement of quick version launch and diversity of application users. Appendix A and Appendix B shows development of website *Red Drop* using two approaches Agile based, and UML based. These can be viewed for differentiating approaches for software applications and web applications.

Having gone through the work presented so far in previous chapters, it can be said that there are several approaches for developing web applications having different techniques available to follow the approach with unique reasons for choosing them as base technique. The factors which affect this selection are complexity of web application, frequency of change of requirements, cost, team management, domain of users etc. Agile development approach forces on quick development concept avoiding need for large documentation. Three methodologies have been discussed based on agile development approach: Web application development using extreme programming, AWDWF, and agile development based on LIFT. The first methodology, development using XP practices, can be used when project is a simple web application with requirement of fast development and a short development team formation. This methodology is also useful when customer wants his continuous involvement with developer team. The active participation by customer reduces development time and requirement fixing cost. Next methodology based on agile approach is AWDWF, which again supports continuous customer participation but only at time of prototype release. This method can be adopted when customer can not involve with developer team in each development phase but can give feedback for each prototype developed by the team. This approach gives some better quality result as development team is kept focused on the core business logic. This method requires a larger team than XP based web application development method. There is one more methodology discussed based on agile approach, LIFT based development, which is more managed development method in compare to previous two. Also, it can be used for somewhat complex applications

provided they do not require very continuous handling of change in requirements. Though, this requirement can also be fulfilled by help of a separate team formed to handle change in requirements only.

Object oriented development approach realizes entities of the project with real world objects thus making possible to develop complex applications with ease of managing them along with easy maintenance work. CORBA is one of the methodologies for developing web applications based on object oriented concepts. Though, it is very old approach but still very effective cause of regular modifications. If web application is very complex and enough man resource is available for development then CORBA is one the good options. Main strength of CORBA along with its object oriented development environment is the services it provides. WebComposition is another object oriented methodology which is again capable of developing much complex web applications in a very cost effective manner.

UML based web application approach provides a very easy and effective way to manage and control development process by use of UML diagrams. The methodology defined in this thesis work based on UML development approach uses extended UML model derived extended from UML 1.3. This methodology can be used when there is a requirement of handling very quick and major changes in requirements very effectively in very less time. The first time development of web application may take some extra time but release of next version is very cost and time effective.

Next in this series is sequential development methodology which has been blended with taste of iterative development approach. This methodology is useful when development is to be done for SMEs. The main advantage of this approach is that completion of each iterative phase gives an executable and deployable module. Thus, if there is a requirement of development of web application which is to be used by SMEs and there is possibility of a large number of risks along with shortage of development team then this method can be applied.

When some model for web application development is already is chosen but there is a need for making some changes in that model to achieve targeted goals with effective and better results, then reverse engineering is best option to go with. The only disadvantage of using this methodology is that final result depends on model chosen: better selection, better process application, better results. Also, there is a need of highly experienced employees for the task because quality degradation in early stage comes out with very worse results.

Next is KITS, a methodology based on knowledge based web application development approach. This methodology can be used when there is non-availability of expert persons and there is requirement for web application development which cannot be delayed and must be initiated at once.

REFERENCES

References

1. Agile development: An overview; http://www.richappsconsulting.com/blog/blog-detail/agile-development-an-overview/; Retrieved 9th April, 2013.
2. Campbell-Kelly, Martin; Aspray, William (1996), "Computer: A History of the Information Machine"; New York: Basic Books. ISBN 0-465-02990-6.
3. Center of online learning, research and service. http://www.uis.edu/colrs/learning/technologies/web20/ . Retrieved 20th May, 2013.
4. D. Ungar and R.B. Smith; "Self: The Power of Simplicity"; Proc. OOPSLA 87, ACM Press, New York, 1987, pp. 227- 242.
5. Dieter Blomme, Nico Goeminney, Frank Gielen and Filip De Turck; "A Two Layer Approach for Ubiquitous Web Application Development"; IEEE 2009: Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns.
6. Douglas C. Schmidt, Steve Vinoski; "Object Interconnections: An Introduction to CORBA Messaging"; SIGS C++ Report magazine, November/December 1998, column 15.
7. Dr. Dobb, The world of software development; http://www.drdobbs.com/; Retrieved 9th April, 2013.
8. Eric J. Byrne; Software Reverse Engineering: A Case Study, Software—Practice and Experience; Vol. 21(12), 1349–1364 (December 1991).
9. Eun Sook Cho, So0 Dong Kim, Sung Yul Rhew, Sang Duck Lee, Chang Gap Kim; "Object-Oriented Web Application Architectures and Development Strategies"; IEEE 1997.
10. F. Ricca; "Analysis, Testing and Restructuring of Web Applications"; IEEE ICSM 2004.
11. Frank Maurer and Sebastien Martel; "Extreme Programming Rapid Development for Web-Based Applications"; IEEE Internet Computing January 2002.
12. G. Kappel, B. Proll, W. Retschitzegger, and W. Schwinger; "Customisation for ubiquitous web applications: a comparison of approaches"; International Journal of Web Engineering and Technology, vol. 1, no. 1, pp. 79 – 111, 2003.
13. Grady Booch; "Object oriented development"; IEEE transactions on software engineering Feb., 1986.
14. Geoffrey Elliott (2004). Global Business Information Technology: an integrated systems approach. Pearson Education. p.87.
15. H. M. Sneed; "Software renewal: a case study"; IEEE Software 1984.
16. Hu Dong, Yu Xue; "Designing and implantation of agile framework based on Lift"; IEEE 2010.

17. Hu Ran, Wang Zhuo, Hu Jun, Xu Jianfeng, Xie Jun; “Agile Web Development with Web Framework”; IEEE 2008.
18. Iman Poernomo; "The Meta-Object Facility Typed"; Proceeding SAC '06 Proceedings of the 2006 ACM symposium on Applied computing.
19. J. Zettel et al.; “LIPE: A Lightweight Process for E-Business Startup Companies Based on Extreme Programming”; Proc Third Int’l Conf. Product-Focused Software Process Improvement (PROFES 2001), Springer Verlag, Berlin, 2001.
20. James Duncan Davidson, Danny Coward, Java Servlet Specification ("Specification") Version: 2.2 Final Release. Sun Microsystems. pp. 43–46.
21. M. Fowler et al.; “Refactoring: Improving the Design of Existing Code”; Addison Wesley Longman, Reading, Mass., 1999.
22. Makoto Yoshida, Noriyuki Iwane; “Knowledge-based Experimental Development of Web Application Systems”; The Fifth International Conference on Computer and Information Technology (IEEE 2005).
23. Manifesto for agile software development; http://www.agilemanifesto.org/; Retrieved 9th April, 2013.
24. McConnell, Steve (1996); Rapid Development: Taming Wild Software Schedules; Microsoft Press, ISBN 1556159005.
25. Mehdi T. Harandi; “Building a Knowledge-Based Software Development Environment”; IEEE Journal on selected areas in communications, June 1988.
26. Methodology:: Development models; http://myprojects.kostigoff.net/methodology/development_models/development_models.htm; Retrieved 14th April, 2013.
27. Moshchuk, S. D. Gribble, and H. M. Levy; “Flashproxy: transparently enabling rich web content via remote execution”; Proceeding of the 6th international conference on Mobile systems, applications, and services. Breckenridge,CO, USA: ACM, 2008, pp. 81–93.
28. Nancy L. Russo, Judy L. Wynekoop, Diane B. Walz; “THE USE AND ADAPTATION OF SYSTEM DEVELOPMENT METHODOLOGIES”; International Resources Management Association, International Conference, Atlanta, Georgia, 1995.
29. Object Management Group; http://en.wikipedia.org/wiki/Object_Management_Group; Retrieved 13th May, 2013.
30. Object oriented modelling language history; http://en.wikipedia.org/wiki/File:OO_Modeling_languages_history.jpg#filehistory; Retrieved 14th April, 2013.

31. OMG; "Catalog of OMG Modeling and Metadata Specifications"; Retrieved 14th April, 2013.
32. P. McKinley, S. Sadjadi, E. Kasten, and B. Cheng; “Composing adaptive software”; Computer, vol. 37, no. 7, pp. 56–64,2004.
33. Paolucci. M, Kawamura.T, Payne.T.R, Sycara.K.; “Semantic Matching of Web Services Capabilities”; International Semantic Web Conference, 2002.
34. Principles behind the agile manifesto; http://www.agilemanifesto.org/principles.html; Retrieved 9th April, 2013.
35. R. Abbott; “Report on teaching Ada”; Science applications, Inc., Dec., 1980.
36. R. N. Britcher and J. J. Craig; ”Using modern design practices to upgrade aging software systems"; IEEE Software 1986.
37. Ritika Maheshwari & Rod Fatoohi; “Design and Implementation of CORBA-Based Subscription Server”.
38. S. Mukherjee, G. Yang, and I. Ramakrishnan; “Automatic Annotation of Content-Rich HTML Documents: Structural and Semantic Analysis”; 2003, pp. 533–549.
39. Sashimi waterfall software development process; http://www.managedmayhem.com/2009/05/06/sashimi-waterfall-software-development-process/; Retrieved 14th April, 2013.
40. Serena; “An introduction to agile software development”; June 2007.
41. Siegel, J.; CORBA Fundamentals and Programming; Wiley, 1996.
42. Software development methodology; http://en.wikipedia.org/wiki/Software_development_methodology; Retrieved 7th May, 2013.
43. Susan Gasson; “The role of methodologies in IT-related organisational change”; Proceedings of BCS Specialist Group on IS Methodologies, 3rd Annual Conference, The Application of Methodologies in Industrial and Business Change, North East Wales Institute, Wrexham, UK; September 1995.
44. The django book; http://www.djangobook.com/en/2.0/index.html; Retrieved 13th May,2013.
45. The scala programming language; http://www.scala-lang.org/; Retrieved 13th May, 2013.
46. The size of the World Wide Web (The internet). http://www.worldwidewebsize.com/. Retrieved 19th May, 2013.
47. UML diagrams overview; http://en.wikipedia.org/wiki/File:UML_diagrams_overview.svg; Retrieved 14th April, 2013.
48. UML Superstructure Specification Version 2.2. OMG. April 2013.

49. Unified Modelling Language, Ver 2.0, OMG, 2005.
50. W. Gellersen, and Martin Gaedke; “Object oriented web application development”; IEEE internet computing 1999.
51. Web application. http://en.wikipedia.org/wiki/Web_application. Retrieved 7th May, 2013.
52. Web development that doesn’t hurt; http://rubyonrails.org/ ; Retrieved 13th May, 2013.
53. Wei Huang, Ru Li, Carsten Maple, Hongji Yang, David Foskett, Vince Cleaver; “Web Application Development Lifecycle for Small Medium-sized Enterprises (SMEs)”; IEEE 2008.
54. Winston Royce; “Managing the Development of Large Software Systems”; IEEE WESCON, 1970.
55. Welcome to AgroUML; http://argouml.tigris.org/; Retrieved 20th May, 2013.
56. Wookjin Lee, Sanghyun Park, Keeyoull Lee, Chunwoo Lee, Byungjeong Lee, Woosung Jung, Taeksu Kim, Heechern Kim, and Chisu Wu.; “Agile Development of Web Application by Supporting Process Execution and Extended UML Model”; Proceedings of the 12th Asia-Pacific Software Engineering Conference (APSEC’05).
57. XML signature streaming profile of XPath 1.0; http://www.w3.org/TR/xmldsig-xpath/; Retrieved 13th May, 2013.
58. Y. Chen, W. Ma, and H. Zhang; “Detecting web page structure for adaptive viewing on small form factor devices”; Proceedings of the 12th international conference on World Wide Web, Budapest, Hungary: ACM, 2003, pp. 225–233.