Web Engineering:  
Software Engineering for Developing Web Applications

Sharad P. Parbhoo  
prbsha004@myuct.ac.za  
Computer Science Honours  
University of Cape Town

15 May 2014

Web systems are becoming a prevalent part of modern society. Associated with this increased usage, there has been a rushed to migrate to the Web, resulting in systems that are not secure, difficult to use and unmaintainable. The lack of adopting a systematic approach to Web application development is a contributing factor of this online entropy. This research paper explores this, and other factors that have resulted in a disorganized Web. Furthermore, it discusses three different methodologies, namely Agile Web Engineering (AWE), Web Semantic Development Method (WSDM) and User Centred Design (UCD), which should be adopted in order for Web-based systems to be developed suitably for the purpose they were intended. Finally, the methodologies are compared to determine which process would be best at developing complex web applications for two types of users. The result is a combination of the AWE and UCD methodologies, which considers all the factors that need addressing when developing Web systems.
# Table of Contents

1. Introduction .................................................................................................................. 3
   1.1. The Rising Popularity of the Web ......................................................................... 3
   1.2. An Explanation of Software Engineering .............................................................. 3

2. Motivation for Web Engineering .................................................................................. 5
   2.1. The Web Crisis ..................................................................................................... 5
   2.2. Simple vs. Advance Web Applications ................................................................ 5
   2.3. Aspects of Web Development .............................................................................. 6
   2.4. Web Engineering as a Solution ............................................................................ 6
   2.5. An Agile Approach ............................................................................................. 7

3. Agile Web Engineering ................................................................................................ 8
   3.1. Overview of AWE ............................................................................................... 8
   3.2. Components of AWE ......................................................................................... 9
      3.2.1. Business Analysis .......................................................................................... 9
      3.2.2. Requirements Analysis ............................................................................... 9
      3.2.3. Design ........................................................................................................ 10
      3.2.4. Implementation .......................................................................................... 10
      3.2.5. Testing ........................................................................................................ 10
      3.2.6. Evaluate ...................................................................................................... 10
      3.2.7. Deploy ......................................................................................................... 10
   3.3. Summary ................................................................................................................ 10

4. Web Semantic Development Method .......................................................................... 12
   4.1. Overview of WSDM ............................................................................................ 12
   4.2. Components of WSDM ...................................................................................... 12
      4.2.1. Mission Statement ....................................................................................... 12
      4.2.2. Audience Modeling ..................................................................................... 12
      4.2.3. Conceptual Design ...................................................................................... 13
      4.2.4. Implementation Design ............................................................................... 13
      4.2.5. Actual Implementation ................................................................................. 13
   4.3. Summary ................................................................................................................ 13

5. A User Centered Design Approach ............................................................................ 14
   5.1. Overview of UCD ............................................................................................... 14
   5.2. Components of UCD .......................................................................................... 14
      5.2.1. Analysis ....................................................................................................... 14
      5.2.2. Design ........................................................................................................ 14
      5.2.3. Implementation .......................................................................................... 14
      5.2.4. Deployment ............................................................................................... 15
   5.3. Summary ................................................................................................................ 15

6. Comparison of Methodologies .................................................................................. 16

7. Conclusion .................................................................................................................... 17

8. Bibliography .................................................................................................................. 18
1. Introduction

1.1. The Rising Popularity of the Web
Over the past two decades, the Internet and the World Wide Web has grown in its ubiquity, popularity and extent of use and is continuing to grow at an exponential rate, surpassing all other technological developments in history (Murugesan & Ginige, 2001). This technology is used to enhance operations making interaction easier within society, and in industries ranging from travel to commerce to education. On the other hand, the advent of the Web connects society to a wealth of information as well as to each other, bridging geographical divides on a global scale.

The demands of Web applications have also increased significantly over the years (Murugesan & Ginige, 2006). In addition to this, recent advances in wireless technologies, smart phones and other portable computing devices have sparked a revolution in mobile Web applications (Murugesan & Ginige, 2006). A combination of the rise in Web use and its increasing application complexity has made the design, development, deployment and maintenance of Web application projects more complex and difficult to manage.

As our dependence on Web-based systems is increasing, their performance, reliability and quality become more important, which is where software engineering plays a critical role. There are aspects of software engineering that contribute to the sustainability of these Web development projects. Before exploring these aspects, a brief introduction to software engineering is required.

1.2. An Explanation of Software Engineering
Software engineering (SE) is an active field of research and has been since the late 1960's. It is defined as,

“The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software” - IEEE Standard Glossary of Software Engineering Terminology

The objective of SE is to provide a disciplined approach to software development to produce systems that
- Are reliable and robust,
- Address the problems they were developed to solve, and
- Are delivered on time and within budget.

New technologies are continuously emerging. Legacy technologies are constantly evolving. These two factors have a direct impact on the way software is
developed, and thus the need for a sound software development methodology has become essential as criteria for successful software projects. These findings serve as the foundation for SE methodologies, which have recently been classified as either Monumental or Agile. Monumental process focuses largely on documentation prior to development whereas Agile methodologies are concerned more with software deliverables than document deliverables (McDonald & Welland, 2001). The principles of Agile development are discussed further in Chapter 7.
2. Motivation for Web Engineering

2.1. The Web Crisis
Web applications are being built with only the primary objective of the application in mind. Several other significant factors are not being considered such as the users’ needs, and issues relating to content management, website design, performance, security, maintenance and scalability, copyright and privacy.

San Murugesan, editor in chief of the IEEE IT Professional, describes some of the issues associated with the low quality Web systems (Murugesan et al, 2000).
- Outdated data
- Irrelevant information
- Difficulty using website
- Slow Web page responses
- Website crashes
- Security breaches

2.2. Simple vs. Advance Web Applications
The ad hoc approach may have been appropriate for simple content presentation, for instance personal blogs. However, there has been a shift with the introduction of the Web 2.0 to more complex systems that contain text, images and other multimedia. Murugesan and his co-authors compare simple and complex Web systems in his paper Web Engineering. A summary can be seen in the table on the following page.

<table>
<thead>
<tr>
<th>Characteristics of Simple and Advanced Web Applications</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Simple Web-based Systems</strong></td>
<td><strong>Advanced Web-based Systems</strong></td>
</tr>
<tr>
<td>Primarily textual information</td>
<td>Dynamic Web pages due to changing information over time</td>
</tr>
<tr>
<td>Static information content</td>
<td>Large volume of information</td>
</tr>
<tr>
<td>Simple navigation</td>
<td>Difficult to navigate and find information</td>
</tr>
<tr>
<td>Limited usefulness</td>
<td>High performance and continuous availability</td>
</tr>
<tr>
<td>Performance not a major requirement</td>
<td>Calls for risk or security assessment</td>
</tr>
<tr>
<td>Easy to create</td>
<td>Requires larger development teams with expertise in diverse fields</td>
</tr>
</tbody>
</table>
2.3. Aspects of Web Development
There is thus considerably more to Web development than visual design and user interface. It involves planning, Web architectures, system designs, testing and performance evaluation and frequent update and maintenance of the system as the requirements and usage grow.

Hence ad hoc development is not appropriate for complex Web systems. This approach could result in serious problems affecting the final system as per the customer specification, the performance, security, maintainability and scalability of the system and the schedule and budget of the project.

More importantly, faulty Web systems potentially impact business operations and transactions. Organizations that rely on the Web cannot afford frequent downtime or stale information. Inconsistent service frustrates users, costing the enterprise heavily in terms of financial loss, lost customers and ultimately, loss of reputation (Murugesan & Ginige, Web Engineering: Introduction and Perspectives, 2006).

Murugesan further explains that although we are faced with these challenges, most Web application development continues to be “ad hoc, chaotic, failure-prone, and unsatisfactory”. He goes on to say that this state of affairs could get worse because Web systems are becoming more complex and require more interaction, both from users and other automated systems.

2.4. Web Engineering as a Solution
With the popularity of the Web reaching new heights, the need for systematic, measurable and repeatable development processes is apparent (Murugesan & Ginige, Web Engineering, 2001). Web developers need to follow best practices, use better development and planning tools and be disciplined in their approach to creating Web systems.

According to Murugesan, Web engineering provides for these needs with a focus on long-term sustainability in developing Web-based systems and applications.

“Web Engineering uses scientific, engineering, and management principles and systematic approaches to successfully develop, deploy, and maintain high-quality Web systems and applications.” - Murugesan et al., 1999

The objective of WE, he says, is to minimize risks, improve quality, maintainability, scalability, all of which lead to a more controlled environment for Web-based application development. The core of Web engineering is to successfully manage the diversity and complexity of Web system development
life cycles, with the goal of identifying and recognizing potential failures before they manifest.

2.5. An Agile Approach
A report by Andrew McDonald and Ray Welland titled *The Agile Web Engineering (AWE) Process*, distinguishes between Web development and traditional software development. Although the two share certain characteristics, the report describes Web projects as typically having
- Short development life cycles times (±3 months)
- Systems that integrate software and data
- Multidisciplinary teams

It further states that in order for Web-based applications to be more successful, Web-based projects need to focus on
- A more thorough requirements analysis
- Better testing and evaluation of Web-based deliverables
- Issues associated with the evolution of Web-based technologies

McDonald and Welland justify an Agile approach to Web development using the above success criteria.
3. Agile Web Engineering

3.1. Overview of AWE

Agile Web Engineering (AWE) is a methodology based around *The Manifesto for Agile Software Development*. The purpose of the manifesto is to promote:

"... better ways of developing software by doing it and helping others do it. Through this work [developers] have come to value:

(i) Individuals and interactions over processes and tools  
(ii) Working software over comprehensive documentation  
(iii) Customer collaboration over contract negotiation  
(iv) Responding to change over following a plan  

That is, while [they] value the items on the right, [they] value the items on the left more."

McDonald and Welland are aware that the developers and the organization play the most important role in the success of a project, considering the methodology used to carry out the project as having a secondary impact. For this reason, they believe that the Agile route lends itself to (i). Furthermore, they highlight the shortfalls of Monumental processes over Agile, indicating that developers are misguided by the purpose of documentation, using the processes not as they were intended by the designers. Points (ii) to (iv) plus 12 other principles are justified through a high level explanation of what AWE tries to achieve.

They go on to discuss in great detail each phase of the AWE methodology. An overview of this, the **Web Semantic Design Methodology** (WSDM) and a **User Centred Design** (UCD) **Approach to Web-based Software Development** make up the remainder of this literature review.
3.2. **Components of AWE**
The AWE methodology is separated into seven phases, with one main deliverable- the Web application itself. Although supporting documentation may be beneficial, it is not required.

![Figure 1: The AWE Development Lifecycle](image)

3.2.1. **Business Analysis**
The objective of the business analysis is to extract a set of problems that must be addressed by the Web application.

The Web application needs to meet clearly defined business objectives for it to have a metric for success. These objectives should be based around gaining a competitive edge in business, or some other equally important advantage.

The greatest problems should be tackled first, allowing for potential changes to be made early on.

3.2.2. **Requirements Analysis**
The requirements phase is two-tiered. Firstly, it indicates what the proposed solution will do, which is termed functional requirements.

Secondly, it describes what constraints are imposed on the solution, which are known as non-functional requirements. These include addressing performance issues, usability concerns, and security vulnerabilities.

Thereafter, the team should start planning tests to determine criteria of what the right product should do.
3.2.3. Design
A high level design, independent of lower level implementation details, is the
core of this phase. Major issues concerned with building a complex Web
application are addressed here. These include scalability, portability and
redesign problems.

It is important to understand that for the above issues to be addressed, great
attention must be given to this phase.

3.2.4. Implementation
An alternate definition describes implementation as design at a low level. AWE
suggests paired programming to build the Web application, as it provides
extensive testing of features from the perspective of different developers.

3.2.5. Testing
A functional assessment of the features is vital in determining whether or not
what has been built thus far satisfies the projects requirements.

Non-functional testing should assess at a minimum
- Application performance
- Cross-browser compatibility
- Scalability for the expected target audience
- Application security

3.2.6. Evaluate
The evaluation plan is guided by the business and requirements analysis. It
considers end-user usability, an evaluation of which often leads to a greater
understanding of the problem space. Helpful feedback is put back into the
analysis phases, and the analyze-design-implement-test-evaluate cycle repeats.

Evaluation has proven to be a time consuming, expensive task, and thus is not
required for every iteration of AWE. This, however, has the implication that the
success of the project is at risk, as there is a greater chance of over- or under-
engineering the product, or the product not meeting the users’ needs.

3.2.7. Deploy
Initially, the Web application is deployed as a trial in real world scenarios. The
overall performance is evaluated, and shortfalls of the system are recorded,
improved upon and after the system is revised, updates are rolled out.

3.3. Summary
AWE provides a detailed, systematic set of steps to support the successful
development, deployment and maintenance of Web applications as per the end-
users request. While it is quite laborious, the objective of such a thorough methodology is meant to identify and correct any misunderstandings, risks and hidden problems. Each iteration should focus on the problems that pose the highest risk to ensure that efforts on previous iterations are not wasted.
4. Web Semantic Development Method

4.1. Overview of WSDM
WSDM, formerly called the Web Site Development Method, offers a systematic, multi-phase approach to Web design. In a paper titled *Semantic Web Development using WSDM* (Plessers et al, 2006) the authors present a methodology comprising five distinct phases, each focusing on a particular aspect of the Web design cycle.

![Figure 2: The WSDM Development Lifecycle](image)

4.2. Components of WSDM

4.2.1. Mission Statement
The mission statement describes the subject of the website, the purpose it must fulfill and it identifies the target users.

4.2.2. Audience Modeling
The targeted users are classified into audience classes, which is described as a group of visitors that have the same knowledge base and functional requirements.

Audience classes can be further classified into subclasses, where the audience is more specific in terms of their requirements. The characteristics and usability requirements are also expressed here.

The model that represents the audience hierarchy, which includes their characteristics and requirements, is called the audience model.
4.2.3. Conceptual Design
A high level abstraction of the content, functionality and structure of the Web system is defined. Conceptual design is divided into Task Modeling and Navigational Design.

The Task Modeling phase is where the content and functionality are defined. The purpose of this phase is to determine the different tasks that different audience classes must perform, considering the requirements formulated during audience modeling. Each task is broken into elementary subtasks. For each subtask, an object chunk is created that formally describes the functionality required of the subtask.

The goal of Navigational Design is to define a conceptual structure of the Web system, thereby modeling how the audience navigates through the system and perform their tasks. Navigation structures are defined for each audience, indicating dedicated navigation structures.

4.2.4. Implementation Design
Conceptual design models are completed with the relevant information required for the actual implementation. The implementation design comprises three sub phases, namely the System Structure Design, Presentation Design and the Data Source Mapping.

The above processes map the conceptual model of the system onto Web pages, taking into consideration design patterns, and the presentation of different information for different audiences, devices, contexts and platforms.

4.2.5. Actual Implementation
The Web system is implemented according to the implementation design, which serves as a detailed project skeleton.

4.3. Summary
WSDM takes a new approach to Web development, relying on various models to determine the best way to represent the data. It allows a web developer to describe the Web application from different perspectives and at different levels of abstraction. Furthermore, it provides a linear, systematic way to develop Web applications.
5. A User Centered Design Approach

5.1. Overview of UCD
UCD is an iterative, audience driven methodology with a large focus on HCI\(^1\). In an article from the Journal of Software Engineering and Applications, *Exploiting User Centred Design Approach and Interactivity in Web-based Software Development*, the author centres his approach around *Uses and Gratification Theory*, which is an accepted theory to understanding mass communication. This theory aims to explain how people use media for their needs and gratification, focusing on the user instead of the actual message (*Siricharoen, 2011*). He classifies the needs of people into five categories; cognitive, affective, personal interrogative, social, and tension-free needs, each need a different motivation for visitors.

There are other HCI components that the author talks about, but these are out of the scope of this literature review.

5.2. Components of UCD
As per the journal article by Sirichareon, UCD is broken down into four practical phases.

5.2.1. Analysis
Analysis activities include user and task analysis, as well as analysis of the business requirements.

Research takes the form of questionnaires, interviews, focus groups and other requirement gathering techniques. These techniques are used throughout the analysis-design-implement iterations to gather information, allowing developers to effectively meet the needs of users.

5.2.2. Design
This phase is one of the most iterative as it starts off with a simple prototype that evolves until all the critical design decisions have been made.

5.2.3. Implementation
The system is implemented and revised in cycles of analysis-design-implement. Each iteration is a step closer toward better matching the user specification.

---

\(^1\) HCI (human-computer interaction) is the study of how people interact with computers and to what extent computers are or are not developed for successful interaction with human beings (*Rouse, 2005*)
5.2.4. Deployment
The Web application is deployed, evaluated and revised in contribution to the long-term sustainability of the product.

5.3. Summary
User Centric Design processes concentrate on the audience throughout the development lifecycle. The core of UCD involves first and foremost considering the prospective audience and considering the goals of the audience when they visit the Web page. Identifying the information that visitors will likely need is important, as this information must be made accessible and understandable.
6. Comparison of Methodologies

Three distinct Web development methodologies have been discussed separately in this paper. This section aims to compare the different methodologies and determine which option is best for developing complex Web applications with different interacting technologies, meant for two types of users. The two types of users are members and visitors.

While the AWE methodology is the most detailed, the trade-off is the relative additional time and effort invested in requirements elicitation and revision. In terms of complex Web applications, the greater the detail, the better. Coupled with close collaboration with the client, it is hard to ignore this as the most methodical approach to a successful Web system.

UCD on the other hand is a flexible approach that puts the user first. A design primarily based around the users needs is intended to provide the best kind of user experience. However, due to the complex relationship between humans and computers, as well as the general nature of humans, this could result in constantly changing requirements.

Both processes are iterative, with UCD following a similar structure to AWE, with a smaller emphasis on testing and evaluation.

WSDM tries to tackle Web development from a different angle, with a relatively high amount of documentation as compared with the other methods. There is a large amount of estimation, as the method attempts to predict what kinds of functions are needed based on an analysis of the targeted audience. It also tries to achieve additional semantic annotation, which is out of the scope of this literature review, but must be mentioned for completeness.

The linear nature of WSDM means it falls short in terms of customer feedback. Much of the effort when developing using this method goes into modeling relationships between objects. This could be a useful tool to integrate different technologies such as a database with a Web development environment. However, the linearity is not favourable for Web development.

A combination of the AWE and UCD should result in the best methodology for Web development of complex applications. The idea is that UCD serves as the basis for which AWE can revolve, as UCD does not consider a host of factors required for successful deployment. This includes factors such as scalability, security, performance, and maintenance.


7. Conclusion

As the Web pervades us, our dependence on it increases. The demand for Web applications has multiplied exponentially in the last two decades, attracting Web development from various professions.

There is a legitimate and growing concern about the ad hoc manner in which Web-based applications are currently being created. Greater sophistication coupled with an increasing complexity of new Web-based applications has resulted in many new challenges that need to be satisfactorily managed or addressed.

Web-based system development is not just graphic design or content development any more. There are a growing number of complex applications, the success of which is dependent primarily on the development team, and secondarily, on how the project is managed.

Evidence suggests that an Agile methodology to Web development is best, as it allows for rapid prototyping, extensive testing and incremental improvements.

Three methodologies were then presented, two following an Agile methodology and the other, a traditional methodology. AWE turned out to be the most detailed, resulting in the best methodology to handle complexity. WSDM and UCD focused mainly on the end-user, which had the merit of producing a system accurately in line with the customer specification. However, the linear nature of WSDM did not leave room for much improvement, and thus UCD was preferred over WSDM.

Overall, the UCD methodology formed the core of the development methodology when required to make a complex Web application. The AWE process was followed to supplement important factors not considered in UCD, such as testing and evaluation.
8. Bibliography


