Requirements Engineering In Current Web Engineering Methodologies

R. Jeyakarthik
Department of Computer Science and Software
Madurai Sivakasi Nadar Pioneer Meenakshi Women's College
Poovandhi, Sivagangai District
Tamil Nadu, India.

Abstract

Web engineering – the discipline concerned with the development of quality web applications has been receiving a lot of attention in the recent era with the increasing number of web projects being undertaken by many organizations. Many methodologies have been proposed in this regard. Requirements engineering – being an integral and crucial phase in traditional software engineering, also necessitates a careful consideration in web engineering as well. The present paper presents an overview of how well the available web engineering methodologies address requirements engineering. The types of requirements, techniques, activities and the level of detail supported by the web engineering methodologies available today are tabulated and this throws light on the need for new methodologies that address requirements engineering in a greater depth for web applications.

1. Introduction

With the spurge in the development of web applications in the recent era, systematic methodologies for the construction of web applications has received significant attention of the IT research community. Several methodologies have been presented by several research groups [33, 18, 31, 9] in the last few years. However, an analysis of most of most of these approaches would reveal that they focus on the design workflow in the life cycle, while other tasks like requirements engineering, tests and quality management are handled with less relevance or not included at all.

In the development of traditional (non-Web) applications both practitioners and process experts regard requirements engineering as the most important phase in the development process since the most common and time-consuming errors as well as the most expensive ones to repair, are errors usually consequence of an inadequate engineering of requirements. Many techniques have been proposed. There are specific ones for the capture of requirements, such as interviewing or storyboarding, techniques for the specification of the requirements, such as scenarios or use case modeling, and for the validation of the elicited requirements, such as prototyping.

Although the relevance of requirements engineering is well known these techniques are poorly applied in the Web engineering field. We stress that on the contrary, web applications require a more extensive and detailed requirements engineering process due to the number of stakeholders involved and due to the diversity of the requirements including among others requirements on the navigation and on the business processes as well as Web usability. It is always an iterative process.

The study performed by Barry and Lang [2] showed that practitioners find web application development difficult and that there is an increasing demand for them to deliver high-quality Web-based software products in-budget and in-time. They urge to find solutions for user-centered approaches which translate users’ navigational requirements into system representations. Modeling techniques that aid in requirements representation and communication will be essential part of the future CASE Tools used in the development of Web applications.

In this paper, we analyze the requirements engineering support provided by the current web engineering methodologies and this is expected to be of great help to web developers to select the appropriate requirements engineering techniques and include them in the development process of Web applications. In addition, it should help in the continuous improvement process of the existing Web methodologies to focus more on requirements engineering, and therefore contribute to improve the quality of the Web applications that are built with these methodologies.

2. Terminologies

Most of the methodologies analyzed and compared in this work provide a classification of requirements. However, the terminology used in these methodologies is not always the same. In order to make the description of each methodology comparable to the others, a general classification of requirements for Web applications is shown previously to the outline of the methods. It is based on the state of the art of Web methodologies.

- Functional requirements are capabilities that a system must exhibit in order to solve a problem. Functional requirements can be sub-classified in:
• Data requirements also known as conceptual requirements, content requirements or storage requirements. These requirements establish how information is stored and administrated by the application.
• Interface requirements (to the user) also known as interaction requirements or user’s requirements. They give an answer to how the user is going to interact with the Web application.
• Navigational requirements represent users’ navigation needs through the hyperspace.
• Personalization requirements also known as customization or adaptation requirements. They describe how a Web application has to (dynamically) adapt itself, depending on the user or environment profile.
• Transactional requirements, also known as internal functional requirements or service requirements, express what the Web application has to compute internally, without considering interface and interaction aspects.
• Non-functional requirements act to constraint the solution, e.g. portability requirements; reuse requirements, usability requirements, availability requirements, performance requirements, etc.

In this section, we only include those web proposals which contain the phase of the requirements handling in the life cycle of their development process. Some of them covered the requirements phase in early versions; others included it only after a revision. Methodologies are outlined chronologically according to the first publication that included requirements specification. The chronological arrangement gives us an idea of how requirements engineering for Web applications has evolved.

3. The requirements engineering approach of current web methodologies

3.1 WSDM: Web Site Design Method

WSDM is a user-centered approach for the development of Web sites that models the application based on the information requirements of the users’ groups [7]. Its development process is divided into four phases:

• User modeling, where users are classified and grouped in order to study system requirements according to each user group,
• Conceptual design, where a class diagram is designed to represent the static model of the system and a navigational model to represent the possibilities of navigation,
• Implementation design, where models of the conceptual design are translated into an abstract language easily to be understood by the computer, and
• Implementation, where the implementation design result is written in a specific computer language.

We focus on the user modeling phase, which is the relevant one for this work. It aims on the identification of the different users’ roles by performing the following two tasks:

• Users’ classification is the identification of the potentials users/visitors of the Web site and their classification according to their interests and navigation preferences. WSDM proposes to analyze the organization environment where the application will be used, and centers the attention on the stakeholders of the business processes supported by the application. In WSDM the relationships between stakeholders and the business process activities performed are graphically represented by conceptual maps of roles and activities.

• Users’ group description is the detailed description of the users’ groups identified in the previous task. The information requirements, functional requirements and security requirements for each user’s group are described with the help of a data dictionary.

The remaining phases in the WSDM process are based on the users’ classification of this first phase.

3.2 SOHDM: Scenario-based Object-Oriented Hypermedia Design Methodology

The SOHDM approach [21] was the first approach stressing the importance of a process that allows the analysts to capture and define the applications requirements. SOHDM has similarities with OOHDM [33] among others, but it proposes a requirement specification based on scenarios.

The following six tasks are performed during the life cycle of SOHDM; for this work, only the first one is relevant:

• Analysis, where requirements are describe using scenarios;
• Object model realization, where a class diagram is built in order to present the static structure of the system;
• View design, which expresses how the system will be presented to the user;
• Navigational design, where a navigational class model is developed in order to express the
possibilities of navigation in the system;

- Realization of the implementation, where Web pages, the interface and also the database are developed; and, finally,
- Construction of the system, where the system is built.

The requirements definition starts on designing a so called context diagram, similar to the data flow diagrams (DFD) defined by Yourdon [38]. To build such a context diagram the analyst has to identify the external entities that communicate with the application, and the events that trigger the communication between these entities and the application. The set of events is specified as a table showing the entities that participate in an event. SOHDM proposes to associate a scenario with each event. Scenarios are graphically represented using a proprietary notation called SAC (Scenario Activity Chart). A scenario describes the interaction process between the user and the application when an event triggers an activity. It specifies the activity flow, objects involved and transactions performed.

SOHDM proposes a process to get the conceptual model of the application out of these scenarios. The proposed conceptual model is represented by a class diagram. The next step in the SOHDM development process is the regrouping of these classes with the objective to obtain a navigational class diagram.

### 3.3 RNA: Relationship-Navigational Analysis

RNA [3] is a methodology that offers a sequence of steps to develop Web applications focusing mainly on analysis. Its phases are:

- **Phase 1** - Environment analysis: the objective is to analyze the audience’s characteristics. Stakeholders of the application are identified and classified in different groups according to their roles (similar to the user modeling phase of WSDM).
- **Phase 2** – Element analysis: in this phase all elements that are of interest to the application are identified, e.g. documents, forms, information, mock-ups, etc.
- **Phase 3** – Meta-knowledge analysis: achieves to build a schema of the application. RNA proposes to identify objectives, processes and operations related to the application, and to describe the relationships between those elements.
- **Phase 4** - Navigation analysis: in this phase, the schema of the previous one is enlarged with navigation features.
- **Phase 5** – Implementation analysis: consists of the identification of how the models described in phase 4 will be produced in a computable language.

RNA only provides some guidelines of the actions to be performed in each phase. Neither modeling concepts nor a notation is proposed, but the RNA approach is one of the methodologies that first focused on the importance of requirements specification in the development process of Web applications. It emphasized the need of the separation between the analysis of conceptual requirements and the analysis of navigational requirements.

### 3.4 HFPM: Hypermedia Flexible Process Modeling

The Hypermedia Flexible Process Modeling (HFPM) presented by Olsina [26] is a wide engineering-based approach, which includes analysis-oriented descriptive and prescriptive process modeling strategies. It includes technical, management, cognitive and participatory tasks. Therefore, HFPM provides guidelines for the planning and managing of a Web project covering the whole life cycle of such a software project. It consists of thirteen phases; for each phase HFPM defines a set of tasks. For the purpose of this work, the most relevant is the Requirements Model whose related tasks are defined as follows:

- **Problem description.** HFPM does not prescribe a concrete technique to perform the problem description, e.g. natural language can be used.
- **Description of functional requirements using use cases.**
- **Data modeling for the identified use cases.** It proposes the design of a class diagram.
- **User interface modeling using sketches and prototypes to be used in the presentation of drafts to the customer.**
- **Non-functional requirements description, such as security, performance, etc.**

HFPM proposes on the one hand a detailed process to handle requirements. On the other hand it does not prescribe specific techniques, which can be chosen freely by analysts and developers.

### 3.5 OOHDM: Object Oriented Hypermedia Design Model

OOHDM is a widely accepted method for the development of Web applications [33], whose first versions focused on design and did not include requirements engineering. The process in OOHDM is divided in four phases producing the following results:

The conceptual model, represented as a class model, is built in order to show the static aspect of the system.
The navigational model consists of a navigation class diagram and a navigation structure diagram. The first one represents the static possibilities of navigation in the system. The second one extends the navigation class diagram including access structures and navigation contexts.

The abstract interface model is developed using a special technique named ADVs [33].

The implementation consists in the implemented code and is based on the previous models.

The capture and definition of requirements were introduced later in OOHDM by Vilain, Schwabe and Sieckenius [36], proposing the use of user interaction diagrams (UIDs). UIDs base on the well known technique of use cases. Use cases are used to capture the requirements but are considered in OOHDM as ambiguous and insufficient for the definition of the requirements that Web applications have, mainly related to the interaction between the user and the system. Therefore, for the specification of the requirements, this approach suggests the refinement of use cases building UIDs, which are used to graphically model the interaction between users and system without considering specific aspects of the interface. The process to get an UID from a use case is described very carefully in the approach.

3.6 UWE: UML-based Web Engineering

UML-based Web Engineering (UWE) is a methodological approach for the development of Web applications based on the Unified Process [17][5]. It is based mainly on the most relevant concepts provided by other methods, but defines a UML notation (UML profile), sticks to the diagrammatic techniques proposed by the UML and defines a systematic and semi-automatic design process [14].

UWE covers the whole life cycle of Web applications and focuses on adaptive applications. It includes a specific requirements engineering phase where requirements elicitation, specification and validation are handled as separate activities of the process. The final result of the requirements capture in UWE is a use case model completed with documentation describing the users of the application, the adaptation rules, the interfaces and the details of the use case relevant for the use case implementation. The latter can be described textually or modeled by UML activity diagrams.

UWE classifies requirements into two groups: functional and non-functional. Functional requirements contemplated in UWE are:

- Content requirements
- Structure requirements
- Presentation requirements
- Adaptation requirements
- User model requirements

Moreover, UWE proposes interviews, questionnaires and checklists as appropriated techniques for the requirements capture, and use cases, scenarios and glossaries for the requirements specification. To validate them, UWE proposes walk-through, audits and prototypes [19].

3.7 W2000

W2000 [1] is an approach that also extends UML notation to model multimedia elements. These multimedia elements are inherited from HDM (Hypermedia Design Model) [12]. The development process of W2000 is divided into three phases: requirements analysis, hypermedia design and functional design. The first one is the most interesting for our survey.

The requirements analysis in W2000 is divided into two sub-activities: functional requirements analysis and navigational requirements analysis. The requirements elicitation starts with an analysis of the different user roles, i.e. the actors which will interact with the application. Every identified actor has his own navigation and functional requirements model. The latter model is represented by a UML use case model. The navigational requirements are modeled in another use case diagram representing the navigation possibilities of the actors. The graphic notation is defined as a UML extension.

3.8 WebML: Web Modeling Language

The Web Modeling Language (WebML) is a high-level specification language for hypermedia applications. WebML follows the style of both, Entity-Relationship and UML offering a proprietary notation and a graphical representation using the UML syntax. This notation is complemented with a set of activities to be performed for the development of Web applications, such as requirements specification, data design and hypertext design [6].

The methodology focuses on requirements collection and requirements specification. It proposes the use of techniques, such as interviewing and analysis of documentation, but restraints from the use of prescriptive checklists for requirements capture. Requirements collection starts with user identification and personalization needs. In addition data requirements and functional as well non-functional requirements are gathered. To note is that navigation or specific hypertext structuring requirements are not treated separately.

Requirements specification (called requirements analysis) consists in a classical use case specification supplemented with a semi-structured textual description. The use of activity diagrams is proposed by this method to express the workflow of complex use cases. A template based description and mock-ups (sketches) are suggested for the specification of the site view and the style guidelines. Finally, acceptance tests are proposed mainly
to check non-functional requirements.

3.9 NDT - Navigational Development Techniques

NDT (Navigational Development Techniques) [10] is a technique to specify and analyze the navigation aspects in Web applications. NDT focuses on the elicitation and specification techniques selected by NDT for the capture and definition of requirements. The requirements analysis workflow in NDT starts capturing requirements and studying the environment applying interviews, brainstorming and JAD techniques. In a second step the system objectives are captured and described. Based on these objectives the system requirements are identified; NDT classifies them into:

- Storage information requirements
- Actor requirements
- Functional requirements
- Interaction requirements
- Non-functional requirements

Interaction requirements are represented by phrases and visualization prototypes. Phrases show how the information of the system is retrieved and are represented by a special language named BNL (Bounded Natural Language) [4]. Visualization prototypes are used to represent the system navigation, data visualization and user’s interaction.

The whole process to elicit and specify objectives and requirements proposed by NDT is mainly based on templates or patterns. In addition, it uses other requirements definition techniques like use cases and glossaries. The NDT approach proposes a different template for each kind of requirement, so requirements and objectives are described in a structured way. Some fields in the templates only accept specific values allowing for a systematic process. The requirements specification workflow finishes with the revision of the requirements catalogue and the development of a trazability matrix which makes the evaluation of whether the specification covers all the possible requirements.

In the context of the NDT project a case tool, named NDT-Tool, has been developed. This tool supports the filling of the templates and automatic extraction of the design results out of the templates.

3.10 Design-driven Requirements Elicitation

The Design-driven Requirements Elicitation is a part of the design-driven process proposed by Lowe and Eklund [25] in order to develop Web applications. It consists of capturing, defining and validating requirements during the design process, i.e. the design activities should be carried out in such a way that the requirements could be handled and managed at the same time. The process is based on prototyping in order to explore possible solutions and problems to be solved. Users and customers define the requirements based on the study of these prototypes. It is an iterative process, which consists of reducing customers and clients’ doubts. The cycle has three phases: evaluation, specification and construction.

This design-driven process was defined based on an exhaustive analysis of “best practices” in the development of Web commercial application. It treats all the requirements in the same manner. The requirements are: content, interface protocol, navigational structure, look and feel, data internal representation, versions, change control, security, content management, control access, efficiency, user monitoring, functionality support, system adaptation, user identification, etc. In the comparison tables of the next section we use the short form DDDP for the design-driven development process.

4. Types of requirements supported by existing web methodologies

The types of requirements supported by the existing web methodologies is tabulated below.

<table>
<thead>
<tr>
<th>Requirements</th>
<th>WSDM</th>
<th>SOHDM</th>
<th>RNA</th>
<th>HFPM</th>
<th>OOHDM</th>
<th>UWE</th>
<th>W2000</th>
<th>NDT</th>
<th>DDDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>User interface req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Navigational Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Adaptive Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Transaction Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Non Functional Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

5. Techniques and Activities of requirements engineering supported by existing web methodologies

The Techniques and Activities of requirements engineering supported by existing web methodologies are tabulated below:

<table>
<thead>
<tr>
<th>Requirements Capture Techniques</th>
<th>WSDM</th>
<th>SOHDM</th>
<th>RNA</th>
<th>HFPM</th>
<th>OOHDM</th>
<th>UWE</th>
<th>W2000</th>
<th>NDT</th>
<th>DDDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>User interface req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Navigational Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Adaptive Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Transaction Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Non Functional Req</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td><strong>Requirements Definition Techniques</strong></td>
<td>WSDM</td>
<td>SOHD M</td>
<td>RNA</td>
<td>HFP M</td>
<td>OOHDM</td>
<td>UWE</td>
<td>W2000</td>
<td>WebML</td>
<td>NDT</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>------</td>
<td>-------</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
<td>-----</td>
</tr>
<tr>
<td>Natural Languages</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td>Yes</td>
<td></td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Glossaries</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Templates</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenarios</td>
<td>SAC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use Cases analysis</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Formal Languages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Prototyping</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Other techniques</td>
<td>Event List</td>
<td>Interfac e Sketches</td>
<td>UIDs</td>
<td></td>
<td></td>
<td></td>
<td>BNL Phrases</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Requirements Validation Techniques:**

<table>
<thead>
<tr>
<th><strong>Requirements Validation Techniques</strong></th>
<th>WSDM</th>
<th>SOHD M</th>
<th>RNA</th>
<th>HFP M</th>
<th>OOHDM</th>
<th>UWE</th>
<th>W2000</th>
<th>WebML</th>
<th>NDT</th>
<th>DDDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review / Walkthrough</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Audit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Matrix of traceability</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Prototyping</td>
<td></td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Other Techniques</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Acceptance Tests</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**6 Degree of Detail**
Another perspective under which a comparative study can be carried out involves the way how requirements engineering approaches are defined. Some methodologies concentrate largely on the development process, others focus on the techniques or on the structure of the results that must be produced. Therefore, we classify the approaches in three categories:

- **process-oriented**, that is, if the approach describes the steps of a process to be followed in order to perform the requirements capture, definition and validation;
- **technique-oriented**, that is, if it describes the techniques to be applied during the process;
- **product-oriented**, that is, if it gives a description of the results which must be produced during the process.

We analyzed the definition of the approaches and evaluated how detailed they are in the description of the process, the techniques and the products. The result of this evaluation is shown in table 3. The evaluation is done separately for each phase of the requirements engineering selecting one value as follows:

- **process-oriented**: the approach clearly describes the steps to follow (+), the process without details (o), or does not indicate any process at all (-)
- **technique-oriented**: the approach clearly depicts the techniques and the way to apply them (+), it enumerates the techniques to apply (o), or it does neither propose any concrete technique nor references any general techniques (-)
- **product-oriented**: the approach clearly describes the structure of the product to be produced (+), it describes the product without detailing its structure (o), or it does not give any indication about the resulting product (-)

### Table 3: Evaluation of Approaches

<table>
<thead>
<tr>
<th></th>
<th>Process Oriented</th>
<th>Technique Oriented</th>
<th>Product Oriented</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSDM</td>
<td>O</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SOHDM</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>RNA</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>HFPM</td>
<td>+</td>
<td>O</td>
<td>+</td>
</tr>
<tr>
<td>OOHDM</td>
<td>O</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>UWE</td>
<td>+</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>W2000</td>
<td>O</td>
<td>O</td>
<td>-</td>
</tr>
<tr>
<td>WebML</td>
<td>O</td>
<td>O</td>
<td>+</td>
</tr>
<tr>
<td>NDT</td>
<td>O</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>DDDP</td>
<td>+</td>
<td>O</td>
<td>-</td>
</tr>
</tbody>
</table>

### 7. Conclusions and future Work

The paper stressed the need of increased focus on requirements engineering for web engineering projects and attempted to study the extent to which the various requirements, techniques, and activities are supported by the existing web development methodologies.

As can be evident, no single methodology “has it all”. The area of requirements engineering for web projects clearly presents enormous opportunities for research in this area that can lead to an “optimal” methodology that can address the maximum level of detail in this crucial area. Such research would be of immense help to organizations and individuals in their endeavor to engineer the best web applications.

### 8. References


Web Applications. In Web Engineering, Kappel G., Pröll B., Reich S., Retschitzeger W. (Eds.), dpunkt verlag (in German).


