REQUIREMENT ENGINEERING

OF A

WEB APPLICATION

A Dissertation Submitted in the Partial Fulfillment for the Award of

MASTER OF TECHNOLOGY

IN

SOFTWARE ENGINEERING

Submitted To	Submitted By
Ms. Abhilasha Sharma	Shashank Mittal
Assistant Professor	2K11/SWE/14
Department of Software Engineering	

Department of Software Engineering

Delhi Technological University

Bawana Road, Delhi – 110042

2012-2013

DECLARATION

I hereby declare that the thesis entitled ***Requirement Engineering of a Web Application*** which is being submitted to the Delhi Technological University, in partial fulfillment of the requirement for the award of degree of Master of Technology in Software Engineering is an authentic work carried out by me.

Shashank Mittal
2K11/SWE/14
Department of Software Engineering
Delhi Technological University
Delhi – 110042

CERTIFICATE

This is to certify that Major Project Report entitled ***Requirement Engineering of a Web Application*** submitted by Shashank Mittal, Roll Number 2K11/SWE/14 is fully adequate in scope as a Major Project submission in partial fulfillment of the award of degree of Master of Technology in Software Engineering, Department of Software Engineering, Delhi Technological University, Delhi.

July 2013

Ms. Abhilasha Sharma

Assistant Professor

Department of Software Engineering

Delhi Technological University

Bawana Road, Delhi- 110042

ACKNOWLEDGEMENT

Completion of Major Project report has been made possible by support, advice and encouragement of many people. I take this opportunity to express my sincere gratitude to all of them.

My guide Ms. Abhilasha Sharma, Assistant Professor, Department of Software Engineering, DTU, played a vital role throughout my project. Her advice, encouragement, valuable suggestions and constant support throughout my work helped me to successfully complete the project report. Without her support, this report would not be the same as presented here.

And I would also like to thank Prof. Daya Gupta, HOD, Department of Computer Science, DTU and all the faculty members of the department for their help providing whenever in was in need during my project.

Shashank Mittal

2K11/SWE/14

M.Tech (Software Engineering)

Delhi Technological University, Delhi- 110042

TABLE OF CONTENTS

LIST OF FIGURE(S)

LIST OF TABLE(S)

LIST OF ACRONYM(S)

EORM: Enhanced Object-Relationship Model

HDM: Hypermedia Design Model

HFPM: Hypermedia Flexible Process Modeling

JAD: Joint Application Development

NDT: Navigational Development Technique

OO-H: Object-Oriented Hypermedia Method

OOHDM: Object Oriented Hypermedia Design Model

QAW: Quality Attribute Workshop

RE: Requirement Engineering

RMM: Relationship Management Methodology

RNA: Relationship Navigation Analysis

SOHDM: Scenario-Based Object Oriented Hypermedia Design Methodology

SRS: Software Requirement Specification

UML: Unified Modelling Language

UWE: UML-Based Web Engineering

WebML: Web Modelling Language

WebRE: Web Requirement Engineering

WSDM: Web Site Design Method

WUML: Web Unified Modeling Language

ABSTRACT

Web applications have insufficient requirement engineering techniques, presented as one of the most important problems which have to be solved when web engineering emerged. Similarly, requirement engineering also has some problems like undefined system boundary and insufficient requirement information. Most of the work that have been done for the development of Web applications, focuses mainly on the design and pays less attention on requirements engineering. Furthermore, traditional techniques for specifying requirements are not appropriate to support distinctive characteristics of Web applications such as Navigation. In this thesis, the discussion will be on the most relevant methods in Web engineering, paying special attention to Web application requirements specification.

CHAPTER 1

INTRODUCTION

INTRODUCTION

Requirement engineering is one of the most important phase in software project development as other phases in the life cycle of software development depends on this activity [41][45]. The complexity of the software systems is rapidly rising. The process of requirement engineering becomes complicate with the raising complexity of the systems. Requirement engineering is responsible to collect all the activities involved in discovering, documenting, and maintaining all the requirements of the systems [44]. Wrong requirements may arise a numbers of consequences such as the product may be delivered late, cost more than the original estimation, customer and consumer may not be satisfied, and the final system may be unreliable and may cause regular system defects.

In present scenario, web is the powerful technology from where people can collect information on any subject and topic. Web is very dynamic in nature whose requirements changes very frequently and the number of stakeholder are very large, so it is very tough to gather exact requirements. And whatever the research is done for web and web application is basically focuses on the design, and less pay attention for requirement engineering.

1.1 Motivation

It is well known that requirement engineering should be effective and efficient. This is very essential because software systems are to meet the expectations of their users and customers, and software is to be delivered on time and within budget [4]. However, when Web Engineering was introduced in late nineties [16], one of the most important problems that was presented to be solved and the problem was the existing requirement specification techniques were not fully sufficient for Web applications.

During last three decades, in the context of the Requirement Engineering for web applications, several approaches for the requirement specification have been presented. Some examples are: Constantine and Lockwood, Jaaksi, Leite's approach, Duran or Rosenberg's approach and Scott, these approaches done great job in the specification of requirements related to the structure and the behavior of a software system. Other approaches such as Chung *et al.*, Cysneiros *et al.*, or Botella *et al*, these approaches have mainly focuses their efforts on defining mechanisms for the description of non-functional requirements in terms of the performance, the reusability, accuracy or the reliability of a software system. However, any of these approaches are not sufficient for the requirement specification of web application requirements.

Web applications are developed over the World Wide Web paradigm in which an aspect that has become a critical problem for web applications, which has been poorly considered in traditional software development [26]. Navigation is encouraged in Web applications because of the focus of Web applications is mainly based on communication [35]. Since the initial goal of the World Wide Web is based on its role as an information medium, generally Web applications are fully or partially developed as brochures or magazines, and their development mainly involves retrieving information (capturing) and organizing information domain (complex information domain) and making that information domain accessible to users. The information is organized according to the hypertext paradigm, in a complex structure made up of nodes, links and anchors, which allow users to navigate throughout the information in a structured and a non-linear way.

The Requirement Engineering activities for these publishing-oriented Web applications require taking decisions that typically depends on an editor's choice rather than on an engineer. The determining organization for this type of information means navigation structure is similar to deciding the structure of a magazine or a brochure. Consider for very large Web applications such as Amazon, Flipkart or Ebay where a major part of these web applications is specifically oriented on to provide users with product information. In the words of Philip Greenspun: *When you put a magazine-like site, you are publishing. Virtually all important decisions that you must make are publishing decisions. Eventually you will have to select technology to support those decisions, but that is a detail*.

In the context of web engineering, Navigational requirements is considered to be a first thing and very important to decide. Several approaches such as OOHDM [42], WebML [22], UWE [35], WSDM [13], W2000 [7], SOHDM, RNA or OOH [23] have been presented which supports the development of Web applications, they provide different development processes in which a main role is played by a navigational model. This model has been introduced to handle the Navigational needs properly. However, in this model Navigation describes at the conceptual level. At the requirement level, little support is provided for describing Navigation [35].

Most web engineering methods prescribe the use of traditional requirements techniques for specification requirements of web application. However, several works such as Burdman, Lowe, England & Finney, and Overmyer have stated that new RE techniques are required for specifying the web application requirements since RE techniques for traditional software are not appropriate in terms of effective and efficient for supporting distinctive characteristics of web applications mainly for Navigation and in relation with this, how Web Requirement Engineering researchers initiatives such as MDWnet are starting to consider the field of Requirement Engineering for the web application has been described. It is not surprising that the survey did by the Cutter Consortium discover that the top problem areas for large-scale Web applications projects were [11]:

- Project schedule delays (79%)
- Budget overruns (63%)
- Lack of required functionality (53%)
- Poor quality of deliverables (52%)

In the last three decades, the development of traditional software, these problems are very similar to those that the Requirement Engineering community has been trying to solve. These problems are mainly occurred due to lack of user information, incomplete requirement specification and variable requirement specification. In a survey did by Macdonald & Welland in 2001, Web application developers stated that most of their Web development projects are running over budget and overtime basically because of two reasons:
(1) Problems in capturing or gathering requirements and,
(2) Poor communication between the developers and their clients.

From the above discussion, it is clear that RE techniques for specification web applications requirements are needed. However, if translation of requirement specifications is not done into the proper software artifacts, the requirement specifications alone are of little use. This is a basic problem that the requirement engineering community has been trying to resolve from its beginning: how to go from the problem space i.e., user requirements to the solution space i.e., design and implementation with an effective and efficient methodological guidance.

1.2 Organization of Thesis

Rest of the thesis is organized as follows:

Chapter 2 describes an overview of related researches done by other researchers in this field. Highlights the major contribution and research work carried out in the field of Web development methodologies.

An overview of requirement engineering is provided in *Chapter 3*; this chapter comprises the importance of requirement engineering and the approaches used for requirement engineering.

In *Chapter 4*, an overview of web has been covered. This chapter also covers the history of web and importance of the web.

How requirement engineering can be helpful in development of web application & how web can be helpful in requirement engineering process has been discussed in *Chapter 5* and *Chapter 6* respectively.

Chapter 7 provides an analysis of the several methodologies of web application development and the comparative study of the different web application methodologies has been done in *Chapter 8* of this thesis followed by conclusion and references.

Finally, in *Appendix A*, case study of requirement engineering of a web application using NDT method has been done.

CHAPTER 2

OVERVIEW OF RELATED WORK

OVERVIEW OF RELATED WORK

It is well known that requirement engineering should be effective and efficient. This is very essential because software systems are to meet the expectations of their users and customers, and software is to be delivered on time and within budget [4]. However, when Web Engineering was introduced in late nineties [16], one of the most important problems that was presented to be solved and the problem was that existing requirement specification techniques were not fully sufficient for Web applications. During last three decades, in the context of the Requirement Engineering for web applications, several approaches for the requirement specification have been presented. The work in this thesis specifies several web development methodologies chronically.

The very first in this thesis is OOHDM (Object Oriented Hypermedia Design Model) was developed in 1996 by Schwabe and Rossi [42]. This was one of the first approaches which provide a methodological solution for development of web application. It is based on object oriented paradigm and emphasizes the separation of the navigational aspect from other such as the conceptual aspect and the interface aspect.

Next methodology that is discussed in this thesis is WSDM (Web Site Design Method) was developed by De Troyer and Leune in 1998 [13]. It is a user-centered approach which defines a web application by describing the requirements of the different groups of users that must interact with it.

SOHDM (Scenario-based Object-Oriented Hypermedia Design Methodology) was developed by Heeseok Lee, Choogseok Lee, and Cheonsoo Yoo in 1998 [26]. This approach considered requirement engineering activities in the development of a web application from its initial version and as its name indicates it is mainly based on the use of scenarios.

After that UWE (UML-based Web Engineering) was developed by Nora Koch in 2006 [35] has been discussed. This is completely based on diagrammatic techniques on UML.

Later on methodology that is specified is Building Web Application with UML was introduced by Jim Conallen in 1999 [31] based on unified process. In this the graphical notation of UML is extended throughout stereotypes.

WebML (Web Modelling Language) was developed by Piero Franternalli and Stephano Ceri in 2000 [39] which is a high level modeling language for web application follows the style of both Entity-Relationship and UML.

After that methodology used is OO-H (Object-Oriented Hypermedia Method) was presented by Oscar Pastor, Cristina Cachero and Jaime Gomez in 2003 [23]. This approach emerged formally as an extension of OO-Method.

After that methodology used is W2000 was developed by Paolo Paolini, Luciano Baresi and Franca Garzotto in 2001 [7] is presented as an extension of UML with web design concepts borrowed from HDM (Hypermedia Design Model).

Next methodology used is NDT (Navigational Development Techniques) was presented by Maria Jose Escalona in 2004 [18] is an approach for the web application development that is focused on requirement and analysis phases.

CHAPTER 3

REQUIREMENT ENGINEERING: PROCESS & PURPOSE

REQUIREMENT ENGINEERING: PROCESS & PURPOSE

Requirement Engineering is the area of software engineering that focuses on understanding customer needs and expectations (requirement elicitation), requirement analysis and specification, requirement prioritization, requirement derivation, partitioning and allocation, requirement tracing, requirement management, requirement verification, and requirement validation [48].

Requirement engineering is one of the important activity in software project development as other phases in the life cycle of software development depends on this activity [41][45]. As the name implies requirement engineering is responsible to collect all the activities involved in discovering, documenting, and maintaining all the requirements of the systems [44]. Wrong requirements may arise a numbers of consequences such as the product may be delivered late, cost more than the original estimation, customer and consumer may not be satisfied, and the final system may be unreliable and may cause regular system defects. According to survey conducted by Standish Group Study, 1994 stated that 13.1 percent projects fail due to incomplete requirements and 8.8 percent projects fail due to frequent change in the requirements. According to the another survey conducted by ESPI in 1995 stated that about 40- 60 percent of all defects found in software projects can be traced back to errors made during the requirements stage. The only way to kill these problems for requirement engineering is to follow the best processes, tools, technologies and practices [32].

According to IEEE standard 610.12-1990 requirement is defined as:

a) Condition needed to solve a user's problem.
b) Condition to be met or possessed by the system to satisfy a formal agreement.
c) Documented representation of conditions as in (a) and (b).

Requirement engineering can be defined by the four processes: requirement Elicitation, Requirement Analysis and Specification, Requirement Validation, and Requirement Management.

3.1 Importance of RE

a) *Complete and Accurate Understanding (Projection) of System:* Success of software development process depends on the proper requirement fetching from the costumer which provide it correct direction to start with. But in practice, often software projects are undertaken without giving requirement phase a proper hit. Developers make decision too early and without understanding of all the constraints on the system. As a consequence unnecessary and (more typically) incorrect assumptions are built into the system and the result is all too often a system which fails to meet customer’s expectations! [17].

b) *Reduced Cost of Detecting and Correcting Errors:* Requirement engineering phase is the very preliminary stage of any software development model which causes to higher error correction cost, if latter on at some stage developers’ team come to know that some requirement has not been modeled correctly or it is missing. One way to understand the importance of requirements is to look at the costs of correcting errors. As an example, the typical relative costs of correcting an error at a particular stage of software development are shown in Table 3.1 [12].

Phase	**Relative Cost**
Requirements	0.1-0.2
Design	0.5
Coding	1
Unit Testing	2
Acceptance Testing	5
Maintenance	20

Table 3.1 Relative Costs of Correcting Errors Made at Requirement Stage of a Project

According to Table 3.1, as an instance the cost of fixing errors at requirement phase is 0.2 times costlier than the cost of fixing errors at coding phase. The percentage of errors in released software due to specifically to requirements to be in excess of 50% of all errors detected [38].

c) *Significantly Reduces the Probability of Project Failure:* If a software project fails, the reasons includes cost, scheduling, quality issues, and/or objective achievement. So, by requirement engineering development team can better estimate cost and schedule, can gather actual objective. Requirement engineering also help in identifying risks that can cause failure, and can categorize them can lead to significantly lower failure rates.

3.2 Approaches

We have a large set of techniques available used in each of the requirement engineering processes. Requirement engineering has three processes: (1) Requirement Elicitation, (2) Requirement Specification and (3) Requirement Validation as shown in Figure 3.1.

Requirement Elicitation is a process of gathering the requirements from the customers and other stakeholders, also called as requirement gathering. Requirement elicitation practices include interviewing, joint application development, brainstorming, concept mapping and etc.

Requirement Specification is a process of describing complete behavior of system to be developed. It describes how user can interact with system. It contains all the functional and non-functional requirements of the system. For requirement specification we have several approaches includes natural language, glossary and ontology, use case modeling, prototypes, etc.

Requirement Validation is a process determining whether the requirements describes in requirement specification document is complete as well as verifiable. The approaches include for requirement validation are audits, reviews and walkthrough, traceability metrics and prototypes for validation.

3.2.1 Requirement Elicitation

A thorough discovery of software requirements for a business is almost never available readily at an analyst's fingertips; can rarely be the requirements can quickly looked up as one would gather information for a paper or study for a test. Much of technical or business requirements are not documented anywhere; it is in the minds of stakeholders, in a feedback that has yet to be obtained from the end users and from a study of flowcharts and surveys that have to be created, so requirements must be gathered, or dragged out and in doing this methodology must be logical and carefully. The importance of requirement elicitation cannot be overstated; it is the most important to any requirements project. One scholarly article notes: *Mistakes made in elicitation have been shown many times to be major causes of systems failure or abandonment and this have a very large cost either in the complete loss or the expense of fixing mistakes [10].* For preventing these types of errors, adequate amount of study and preparation for elicitation can go a long way. Therefore, the purpose of requirement elicitation is to identify risks, business needs, and assumptions associated in a given project.

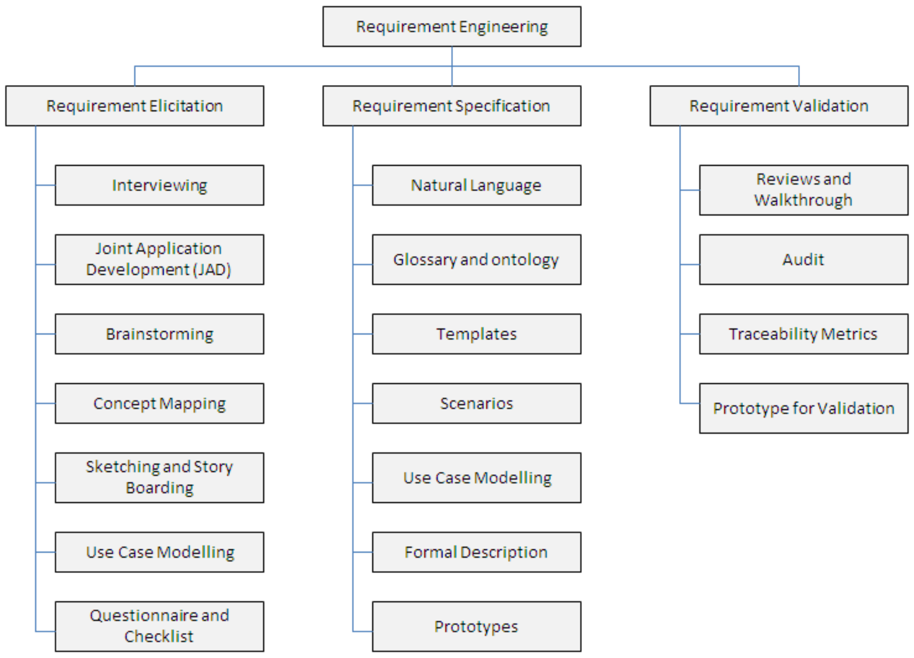


Figure 3.1 Techniques for Requirement Engineering Phases

Steps to Prepare for Elicitation

1. The first step in requirement elicitation is collecting a detailed and accurate understanding of the business need. During the requirement elicitation process, strong understanding of the business need of an analyst will help in selecting the proper stakeholders and elicitation techniques.
2. An analyst's next step in eliciting requirements is ensuring that an adequate amount of stakeholders are secured for the project's duration. For, Business Analysis Body of Knowledge, the definitive guide to all things related to business analysis notes; a good analyst must *actively engage stakeholders in defining requirements* [10]. A project's stakeholders may include customers/end users, suppliers, project manager, project sponsors, quality analysis, regulators, operational support, domain subject matter experts, and implementation subject matter experts. An analyst must recruit the participation of appropriate stakeholders based on the unique business needs of project. After an analyst has identified and recruited the stakeholders, and chosen the method by which she will elicit requirements outlined below, it is advisable to schedule the time for conducting those methods with stakeholders as far in advance as possible to ensure adequate participation on their parts.

Requirement Elicitation Techniques

After securing the proper stakeholders, an analyst must determine the best techniques for eliciting requirements. Commonly used requirement elicitation methods include:

- *Interviewing***:** Interviewing is a traditional and frequently applied technique. By means of interviews analysts are able to understand the problem and get information about the objectives of the application to be developed. The interviewing technique and certain guidelines of how to use them correctly are described in detail by Durán, Bernáldez, Ruíz and Toro [12] as well as Pan, Zhu and Johnson [46]. Basically, the interviewing process covers four steps: the identification of stakeholders for the interview, the preparation of the interview, the interview itself and the documentation of the results in form of an interview protocol. Interviews are not easy to perform; they require a vast experience of the interviewer who needs to have the ability to choose the most suitable interviewees [46]. One-on-one interviews are among the most popular types of requirement elicitation, and for good reason: they give an analyst the opportunity to discuss in-depth a stakeholder's thoughts and get perspective on the business need and the feasibility of potential solutions. *Research has found that interviews are the most effective way of eliciting requirements* [46]. Whether an analyst chooses to have a structured (with predefined questions) or unstructured interview (with free-flowing, back-and-forth conversation), she must fully understand the business need in order to conduct a successful interview. It is a good practice for an analyst to share interview notes with the interviewee afterward to ensure there were no misunderstandings and to jog the interviewee's thoughts for any further insights.

- *JAD (Joint Application Development):* JAD can be regarded as an alternative to interviewing. It is a group technique that requires the participation of all stakeholders of a project, i.e. analysts, designers, users, system administrators and customers [40]. The requirements are captured in a set of sessions over several days. In each session, the high level requirements are analyzed and the problem field and the documentation are established. During each session the group discusses about the different topics, drawing and documenting, as a result a set of documented conclusions. Such conclusions drive the specification of the system requirements. JAD is based on four basic principles: group dynamic, the use of visualization techniques to improve communication, the support of an organized and rational process, and a philosophy of documentation of type *What You See Is What You Get* [40]. On every JAD session the requirements of the system are becoming more concrete. This technique provides several advantages compared to interviewing mainly because it saves time. In JAD it is not necessary to compare customer's opinions with one another. Conversely, JAD needs a good integrated and organized group of stakeholders.

- *Brainstorming:* Brainstorming is also a group meeting technique similar to JAD. The purpose of gathering your stakeholders for brainstorming is *to produce numerous new ideas and to derive from them themes for further analysis* [48]. An analyst should try to secure a representative from each participating stakeholder group in the brainstorming session. If an analyst serves as facilitator of a brainstorming session, she must ensure that while participants feel free to propose new ideas and solutions, they remain focused on the business need at hand and do not engage in scope creep, gold plating, or become distracted with other business issues [48]. All ideas must be recorded so that they are not lost. The brainstorming method is particularly useful if your project has no clear winning choice for a solution, or if existing proposed solutions are deemed inadequate. The brainstorming process and the resulting follow-up analysis will help ensure that the best possible solution is reached for any business objective. It consists of collecting non-evaluated ideas and information of all stakeholders of the project [48]. The number of participants of such brainstorming meetings should not exceed 10 stakeholders of the project; one of them has to assume the role of moderator, but should not control the session.

  In contrast to JAD, brainstorming is easier to use as it requires less work in the group. Moreover, as brainstorming often provides a better overview of the system requirements, it is frequently used in first meetings where concrete details are not still needed.

- *Concept Mapping:* Concept Mapping is a technique by means of which concept maps are built [46]. Concept maps are graphs with vertexes representing concepts and edges representing relationships between these concepts. These graphs, developed by the project team together with the customer and/or final user, are frequently used as a simple communication medium, mainly because they are written in the customer's language [46]. However, some care is required to avoid a subjective and ambiguous description of complex systems. It is recommended to provide an additional textual description.

- *Sketching and Storyboarding*: Sketching and Story boarding is a technique frequently used by graphical designers in the development of Web applications. It consists of schematic representation (usually on the paper) of the different user interfaces (sketches). These sketches can be grouped and connected using links, building this way a so called storyboard that gives an idea about the navigation structure.

- *Use Case Modeling*: Use Case Modeling is a technique which was developed to define requirements [32] more than for capturing them. A use case model consists of actors, use cases and relationships between them [52]. It is used to represent the environment by actors and the scope of the system by use cases (functional requirements). An actor is an external element to the system (e.g. a user, another system) that interacts with the system as a black box. A use case describes the sequence of interactions between the system and its actors when a concrete function is executed. An actor can take part in several use cases and a use case can interact with several actors. The main advantage of use case modeling is that a use case model is easy to be understood by the user or the customer as well as by the developers. However, sometimes they are not concrete or detailed enough [36, 15]. Thus, they can be supplemented with textual information or another technique like activity diagrams.

- *Questionnaire and Checklist*: Questionnaire and Checklist is a technique that consists of preparing a document with questions for which only short and concrete answers or even with a limited choice of answers (checklist) is possible [32]. The questionnaire can be completed during an interview or it can be used to get information independently from an interview. The drawback of this technique is that the analyst needs certain knowledge about the problem domain and the application to be built in order to prepare the questionnaire and checklist. While they preclude the opportunity for in-person, ad hoc conversations, and surveys are useful for quickly gathering data from a large group of participants. Because free online survey software is readily available, surveys are an inexpensive way to gather objective input from customers or potential end users [15]. As with selecting stakeholders, a successful survey or questionnaire must have well-chosen participants. As one researcher notes, questionnaires *can be useful when the population is large enough, and the issues addressed are clear enough to all concerned* [52]. Surveys can be structured to offer a series of finite choices for feedback, or they can offer open-ended input, depending on the needs of the project at hand. Open-ended surveys are useful for a broader discovery of business needs; however, the larger the number of participants in open-ended surveys, the more prohibitive they are to analyze. Survey wording must be unambiguous and precise. It is good practice for an analyst to politely request that survey participants respond by a reasonable deadline and that they keep any proprietary business information contained within the survey confidential.

- *Terminology Comparison:* Terminology Comparison is a technique that does not resolve the problem of requirement elicitation on its own [15]. Instead, it is a complementary technique used to overcome the communication difficulties, that may arise among developers and users, who do not use the same language. The comparison is used to get a consensus about the terminology which will be used in the project. Therefore, it is necessary to identify those words used for the same concept (correspondence), similar words to express different concepts (conflicts) or if there is not exact concordance in the vocabulary or concepts (contrast) [46].
  Requirement engineering community has proposed many other techniques to capture requirements, such as the analysis of similar systems or documentation.

- *Document Analysis***:** Document analysis involves gathering and reviewing all existing documentation that is pertinent to your business objective or that may hold data related to a relevant solution [32]. Such documentation may include, business plans, market studies, contracts, requests for proposal, statements of work, memos, existing guidelines, procedures, training guides, competing product literature, published comparative product reviews, problem reports, customer suggestion logs, and existing system specifications, among others. In other words, virtually anything that is written related to the project may be useful. This type of elicitation is especially useful when the goal is to update an existing system or when the understanding of an existing system will enhance a new system [46]. However, document analysis alone is rarely enough to thoroughly extract all of the requirements for any given project.
- *Focus Group*: Focus Group consists of a mix of pre-qualified stakeholders who gather to offer input on the business need at hand and its potential solutions [52]. Focus groups are particularly helpful when key stakeholders are not particularly imaginative or forthcoming; a few more vocal stakeholders may help them think through and articulate solutions. Focus groups are also a good way for time-pressed analysts to get a lot of information at once. They may be conducted in person or virtually [10]. Key project sponsors or business owners should still be interviewed individually for thorough discovery.
- *Interface Analysis*: Interface Analysis carefully analyzes and deconstructs the way that a user interacts with an application, or the way one application interacts with another. A thorough interface analysis will describe the purpose of each interface involved and elicit high-level details

about it, including outlining its content [48]. This type of elicitation is essential for software solutions, which almost always involve applications interacting with one another and/or users interacting with applications. But, interface analysis can also be useful for non-software solutions (such as defining deliverables by third parties).

- *Observation (Job Shadowing)*: Observation is quite helpful when considering a project that will change or enhance current processes. Two basic types of observation are available to an analyst: (1) passive observation, where the analyst merely watches someone working but does not interrupt or engage the worker in any way [10], and (2) active observation, where an analyst asks questions throughout the process to be sure she understands and even attempts portions of the work [10]. The nature of an analyst's project will dictate the level of detail an observation should encompass. As with interviews, it is a good practice for an analyst to provide notes from observations and/or a verbal description of understanding of the work for the worker to review in order to be sure that there were no misunderstandings of the process.
- *Prototyping*: Prototyping (storyboarding, navigation flow, paper prototyping, screen flows) is especially valuable for stakeholders such as business owners and end users who may not understand all of the technical aspects of requirements, but will better relate to a visual representation of the end product [48]. Stakeholders often find prototyping to be a concrete means of identifying, describing and validating their interface needs. The prototyping process is normally iterative, improving as more input and evaluation are gleaned from stakeholders. Prototyping may be an interactive screen (normally consisting of hypertext only with no real data behind it), a mock-up (such as a PowerPoint), a navigation flow (such as a Visio diagram), or a storyboard. Simple, throwaway prototypes (such as pencil sketches) may be done in the initial stages of discovery, and more detailed, interactive prototypes may be done once business requirements have been identified [52]. At the latter, more detailed prototype stage, prototype features must fulfill previously identified business needs as outlined in the requirements.
- *Requirements Workshops*: A Requirements workshop involves gathering a previously identified stakeholder in a structured setting for a defined amount of time in order to elicit, refine, and/or edit requirements. To be successful, requirements workshops must include a recorder (or scribe) to record participants' input, and a facilitator to direct the discussion. Because participants' may also brainstorm together and listen to each other's input, they can provide immediate feedback and refinements to identified business needs, which can ensure a fast, effective elicitation of requirements.

3.2.2 Requirement Specification

Requirement specification for a software system is a complete description of the behavior of a system to be developed and may include a set of use cases that describe interactions the users will have with the software. In addition it also contains non-functional requirements. Non-functional requirements impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

Software requirement specification document enlists all necessary requirements that are required for the project development. To derive the requirements, one need to have clear and thorough understanding of the products to be developed. This is prepared after detailed communications with the project team and customer.

The SRS may be one of a contract deliverable Data Item Descriptions or have other form of organizationally-mandated content. An example organization of an SRS [29] is as follows

- Introduction
    - Purpose
    - Definitions
    - System Overview
    - References
- Overall Description
    - Product Perspective
        - System Interfaces
        - User Interfaces
        - Hardware Interfaces
        - Software Interfaces
        - Communication Interfaces
        - Memory Constraints
        - Operations
        - Site Adaptation Requirements
    - Product Functions
    - User Characteristics

    - Constraints, Assumptions and Dependencies
- Specific Requirements
    - External Interface Requirements
    - Functional Requirements
    - Performance Requirements
    - Design Constraints
        - Standards Compliance
    - Logical Database Requirement
    - Software System Attributes
        - Reliability
        - Availability
        - Security
        - Maintainability
        - Portability
    - Other Requirements

Requirement Specification Techniques

After securing the proper stakeholders, an analyst must determine the best techniques for requirement specification. Commonly used requirement specification techniques include:

- *Natural Language:* It is an ambiguous technique to define requirements. Requirements are described in natural language without any kind of rules. Although this procedure is often criticized, it is quite often used in practice.
- *Glossary and Ontology*: Glossary and Ontology are used to define the terminology that should be used in every software project where stakeholders with different background work together. This aspect is critical in the development of Web applications as the development team is usually an interdisciplinary one [36]. Therefore, many methodologies propose the use of a glossary in order to define and maintain the most important and critical concepts related to the application. If ontology is defined, it means that in addition to concepts, the relationships between these concepts are specified. None of the methodologies for the development of Web applications described in this work propose the use of ontologies.

- *Templates:* They are used to describe the objectives and requirements using natural language, but in a structured way. A template is a table whose fields have a predefined structure and are filled in by the development team using the user's terminology. Templates are also known as patterns and less ambiguous than descriptions in natural language due to their structure. However, if templates are too structured they could be difficult to fill and maintain.

- *Scenarios***:** Scenarios consist of the description of the characteristic of the application by means of a sequence of steps [17]. Scenarios can be represented in different ways: as texts or in a graphical form, e.g. by use cases [55]. The analysis of such scenarios provides important information about the requirements of the application [41]. Scenario notations are integrated in many object-oriented analysis techniques.
- *Use Case Modeling***:** Use Case Modeling has been widely accepted as a technique to define requirements although it is also used in requirement eliciting as described in the previous section. However, it has the disadvantage that it is ambiguous when defining complex requirements [36, 15]. For this reason, some approaches that define use cases propose to add a textual description using templates or a more detailed diagrammatic representation [19, 36].
- *Formal Description***:** Formal description is another important group of techniques that proposes in contrast to natural descriptions the use of formal languages to specify requirements. Algebraic specifications for example, have been applied in software engineering for some years. However, they are difficult to be used and understood by customers. Its main disadvantage is that they do not facilitate the communication between customer and analyst. Conversely, it is the least ambiguous requirements representation allowing for automatic verification techniques.
- *Prototypes***:** Prototypes are a valuable tool for providing a context within which users are able to better understand the system they want to be built. There is a wide variety of prototypes that range from mock-ups of screen designs to test versions of software products. There is a strong overlap with the use of prototypes for validation.

- *Quality Attribute Workshop (QAW)*: QAW advocates that concrete quality attribute requirements can be described in the form of quality attribute scenarios and Stakeholders are the best carriers of the different perspectives manifested in quality attribute requirements.

Results in prioritized set of quality attribute requirements that are candidates for architectural drivers

  - QAW Presentation and Introductions
  - Business and Mission Presentation
  - Architectural Plan Presentation
  - Identification of Architectural Drivers
  - Scenario Brainstorming
  - Scenario Consolidation
  - Scenario Prioritization
  - Scenario Refinement

- *Global Analysis:* Global Analysis focuses on identifying and analyzing the factors that have global influence on the architecture. It provides guidance for a classification scheme for factors
  - Technological; e.g. software technology, architecture technology, standards
  - Organizational; e.g. development schedule, budget
  - Product factors; e.g. functional features, user interface, performance

  Global Analysis meant to complement risk and requirement analysis, which can be performed, using other techniques, starts before architectural views are defined and continues throughout the development process. Global Analysis results in issue cards with a list of influencing factors and a discussion of strategies to address the issue.

- *O'Brien:* O'Brien provides guidance for linking architectural decisions to measurable quality attributes that flow from explicitly capturing business goals. Aims to capture deriving quality attributes from the business case, e.g. longevity: product lifetime. It advocates defining and monitoring for visibility, measure of design error for observable outcome.

3.2.3 Requirement Validation

Once requirements are defined, they have to be validated. Through requirement validation the requirement specification is checked to correspond to the user's needs and the customer's requirements [41]. Only few approaches provide techniques to validate requirements. Most of them only define some guidelines about how developers and customers should review the requirement specification in order to find inconsistencies and mistakes. The following is an overview of techniques that are appropriate for requirement validation:

- *Review or Walk-through*: Review or Walk-through is a technique which consists in reading and correcting the requirements definition documentation and models. Such a technique only validates the good interpretation of the information. The verification of documentation inconsistencies and the detection of missing information require more sophisticated methods.
- *Audit*: Audit consists of a check of the results presented in the review documentation. The results are compared with a checklist predefined at the start of the process. It provides only a partial review of the information and results.
- *Traceability Matrix*: Traceability Matrix consists of a comparison of the application objectives with the requirements of the system [12]. A correspondence is established between objectives and how they are covered by each requirement. This way, inconsistencies and non-covered objectives will be detected.
- *Prototyping for Validation*: Prototyping for validation is a technique that consists in building tools based on the requirement specification, i.e. the developers' interpretation of the systems requirements. These prototypes usually only implemented a partial set of functional requirements but provide a global vision of the user interface [3]. In order to use this technique the user has to understand that what he is observing is only a prototype and it is not the final system.

CHAPTER 4

WEB: INCEPTION & USES

WEB: INCEPTION & USES

The World Wide Web (or simply Web), is a system of interlinked hypertext documents accessed on the Internet. One can view web pages with the help web browser that may contain text, images, videos, and other multimedia documents, and provide navigation between pages via hyperlinks.

The Web was invented by Tim Berners-Lee also known as *TimBL*. He made a proposal for an information management system in March 1989 [9]. In December 1990, Robert Cailliau and a young student at CERN helped TimBL in successful implementation of the communication between a HTTP client and server on internet.

4.1 Uses

In modern times, web is the most useful technology which helps people not only in their personal lives, but also in their professional lives. The web helps in achieving this in several different ways.

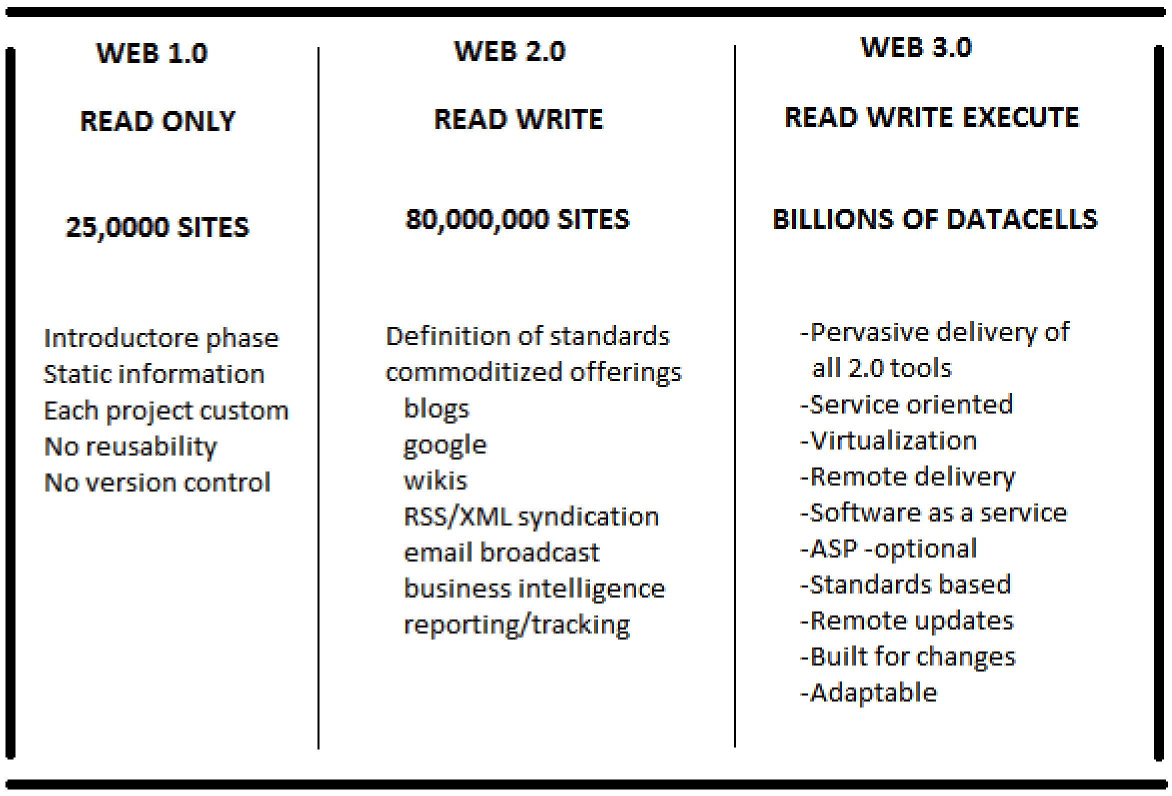
For students and educational purposes the web is widely used in collecting information for the research purpose or for gaining the knowledge of any subject and topic. And for the business personals and the professions like engineers, doctors use the web to filter out the necessary information for their use. The web is the largest encyclopedia for everyone.

In maintaining contacts with the loved one living very far, the web has served to be more useful in communication. Web chatting and emails are the easiest way of communication and the most common for maintaining contacts and relations with the friends and relatives around the world.

Web is also useful for enjoyment and fun purposes these days. As it may be the games, movies, songs, quizzes, videos, conferences or the dramas. Web has provided the users to put an end to their boredom from their lives.

Users to download extremely large number of software enables using web, it is also useful in up gradation of the web and may enables user to work on the project and documentation works using those software, making it much easier and economic than buying the costly software cds and dvds [57].

4.2 History of Web



WEB 1.0	WEB 2.0	WEB 3.0
READ ONLY	READ WRITE	READ WRITE EXECUTE
25,0000 SITES	80,000,000 SITES	BILLIONS OF DATACELLS
Introductore phase	Definition of standards	-Pervasive delivery of all 2.0 tools
Static information	commoditized offerings	-Service oriented
Each project custom	blogs	-Virtualization
No reusability	google	-Remote delivery
No version control	wikis	-Software as a service
	RSS/XML syndication	-ASP -optional
	email broadcast	-Standards based
	business intelligence	-Remote updates
	reporting/tracking	-Built for changes
		-Adaptable

Figure 4.1 Basic Introduction of Web Version

Web 1.0

Web 1.0 referred as the web which was introduced to the public in 1993 and was the initial stage of World Wide Web. Web 1.0 was an early stage of the conceptual evolution of the World Wide Web; users may only view webpages but cannot contribute anything to the content of the webpages. Content providers were few in Web 1.0 and users simply acting as consumers of content [6]. Technically, Web 1.0 does not have the functionality of external editing the webpage's information. Thus, information was not dynamic, can only be updated by the master of that webpages. Economically, web concentrate on the most visited webpages and the software’s cycle releases for the revenue [36]. Technologically, Web 1.0 concentrated on presenting, not external editing and creating so that user-generated content was not available.

Web 2.0

The term Web 2.0 was coined in 1999 and also known as social web, to describe web sites that use technology beyond the static pages of web 1.0 websites. Because of the O'Reilly Media Web 2.0 conference, it is closely associated with Tim O'Reilly which was held in late 2004 [37] [36]. Web 2.0 is a new version of the web, it refers to increasing number of software developers and end-users in the way they uses the Web.

People may interact with and collaborate with each other using web 2.0 in a social media dialogue, where user may treat as a creator of content in a virtual community, in other words websites where people are limited to the passive viewing of content. Web 2.0 includes social networking sites, blogs, wikis and video sharing sites such as facebook, twitter, youtube [53].

Web 3.0

The term Web 3.0 was coined by Tim Berners-Lee, the inventor of the World Wide Web [8] also known as *The Semantic Web*. He defines the Semantic Web as *a web of data that can be processed directly and indirectly by machines.* The Semantic Web is a collaborative movement led by the international standards body, the World Wide Web Consortium [47]. According to the original 2001 *Scientific American* article by Tim Berners-Lee, describes about an expected evolution of the existing Web to a Semantic Web [8], but yet to happen. According to the W3C, *The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries.* [46]

The aim of semantic web is to convert the current web's unstructured and semi-structured documents into web of data in structured format.

Web 1.0	**Web 2.0**	**Web 3.0**
The Web	The Social Web	The Semantic Web
Read only content and static HTML website	User generated content and read write content	Meaningful, portable personal web
Push technology	Share technology	Live technology
No user interaction	User interaction present	User interaction present

Ecosystem	Participation	Understanding itself
Pushed web, text/graphics based, flash	Two way web, blogs, wikis, sharing, podcast, video, personal publishing 2D portals and social networks	The real time, co-creative web, Growing 3D portals, MUVEs, avatar representation, interoperable profiles, integrated games, education and business. All media in and out of virtual worlds.
Search engines retrieve macro contents. Search is fast but most of the times irrelevant results.	Search engines retrieve tags with micro content. Everything: pictures, links, events, news, blogs, audio, video etc. are tagged.	Search engines will hopefully retrieve micro content texts which were tagged automatically.

Figure 4.2 Tabular Description of Difference between Versions of Web

CHAPTER 5

REQUIREMENT ENGINEERING FOR WEB

REQUIREMENT ENGINEERING FOR WEB

In this chapter, what are the drawbacks in the web development processes that we are facing using traditional technique to develop a web application has been discussed and how to can overcome those problems using requirement engineering.

5.1 Drawbacks in Web Development

a) *Less Effective Communication:* Web is very diverse, and for this type of fully diverse environment, communication between all stakeholders is such a difficult task and any small changes may become reason for redundant and unclear requirement. So communicating all stakeholders and then executing the planned activity is less effective.

b) *Problem in Tracing Requirements:* The biggest point of requirement traceability problem is the inability to trace the human sources from where the actual requirements and related information can be gathered [24]. And as web development process continues, there some new requirements usually introduced, and then to check whether these newly introduced requirements is conflicting or not with present requirements, a communication required with the human sources providing requirements. To solve this problem, link should be established with the individual human sources rather than with the groups or community, but in case of web applications this may be a threat of one cause of diversity and unavailability of the application users [5].

c) *Classification of User:* Actually this is the strength for web development that effective classification of users can be done but this strength may turn to threat if done imprudently. While creating web application, it is important to classify the users in groups properly so that complete set of requirements can be gathered as per the priorities of the functionality. Assigning weightage to requirements gathering process should be prudently implemented for each user group otherwise the strength may turn to threat.

d) *Designing Architecture is Very Complex:* In web application development, making requirement specification process easier and effective for the users' point of view, the task to build architecture becomes very difficult and complex. Techniques uses for requirement specification such as use of natural language, sketching, mock ups makes requirement specification process easier but making process of design becomes complicated.

e) *Difficult to Convince Method:* It is difficult to convince the methods that experiences failures or delays encounter to apply in another environment [5].
f) *Difficult to Predict Cost:* One part of this problem is that customers do not know the complete requirement of the web application and how much detail work is necessary calling it complete application.
g) *Difficult to Collect Complete Requirements:* It is difficult to define that what the finished web application needs to look like and how it should operate. Due to large number of stakeholders it is very difficult to stick to one complete set of requirement which is finalised.
h) *Requirements may Preclude Changing to More Suitable Options:* Web is ever changing technology, so as web changes the requirements of web application also changes frequently and need of particular functionality also varies. It is difficult to stick to one set of requirement and latest technology [54].

5.2 How Requirement Engineering can Overcome the Web Application Development Problems

While developing web applications, there are some issues regarding web which makes development of web application different from traditionally system applications.

One main issue is number of web users, under this stakeholders, customers, users, graphical designers, security experts, etc., comes as web users and requirement sources.

Second main issue is variation in requirements, as large number of users demands new requirements satisfying their needs in their own way and satisfying each user in its own navigational and usability perspective is not a possible task. These are two main reasons that emphasize on need some special requirement engineering techniques for web which could handle these issues efficiently. Since the origin of web applications, requirement techniques are introduced especially for web application very frequently.

Requirements for web applications can be classified in following manner as shown in Figure 5.1.

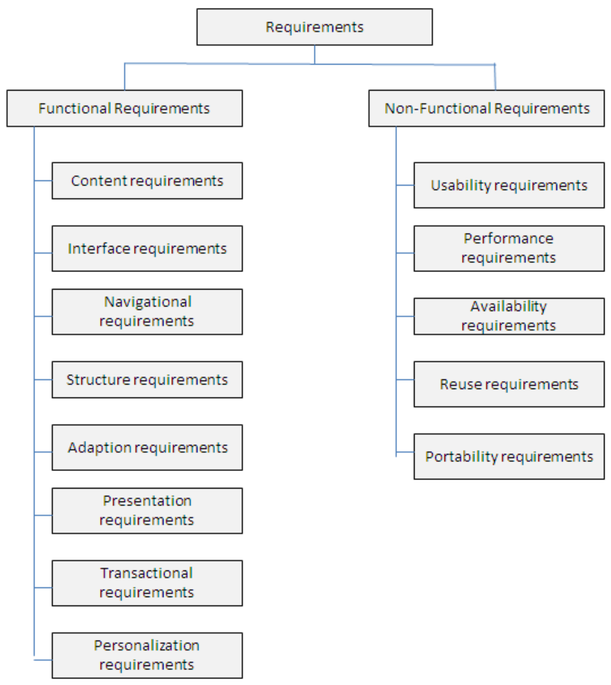


Figure 5.1 Requirements for Web Applications

Key points how requirement engineering works for web application development

- Helps in understanding the web application overall functionality
- Collects operational environment including business objectives.
- Clearly identifies stakeholders (users, customers that need the system and those who fund development).
- Specifies overall technical and nontechnical requirements of web application.
- Develops design architecture of web application.
- Divide overall architecture in subprojects and if the subprojects are also complex then divide them further till they become manageable set of tasks.
- Provide effective mechanism to manage web application evolution. As per requirement repeat the parts of it.
- Resolve nontechnical issues such as human resources development, legal and organizational policies.
- Make development process manageable.
- Measures system's performance and helps in refine and update system [3].

CHAPTER 6

WEB FOR REQUIREMENT ENGINEERING

WEB FOR REQUIREMENT ENGINEERING

In this chapter, what are the drawbacks in the requirement engineering processes that we are facing using different techniques to develop a system and what can be done to overcome those problems taking help of web has been discussed.

6.1 Drawbacks of RE

Requirement engineering process has following drawbacks that are discussed for each sub process of requirement engineering.

6.1.1 Problems of Requirement Elicitation

Issues of requirement elicitation can be classified and grouped into three categories [10]. These are as follows:

a) Issues of Scope,
b) Issues of Understanding User Needs, and
c) Issues of Volatility.

Another term for these scopes by the problems can be used on *Analyzing the Problem* and *Understanding the User Needs* [14].

Issues of Scope

- Boundary of system is not well-defined.
- The requirements may have too little or too much information.
- May give unnecessary design information

Issues of Understanding User Needs

- Issues of understanding between groups and within groups such as consumer and developer
- Customers have incomplete understanding of their needs
- Customers have little understanding of computer capabilities
- Customers have little understanding of limitations
- Analysts may have poor knowledge of problem domain

- Customer and analyst may speak different languages
- Omitting obvious information
- Views of different stakeholders may conflict
- Requirements are often vague and may be untestable. e.g., user friendly and robust

Issues of volatility

- The changing nature of requirements.
- Requirements evolve over time [49].

Requirement elicitation is complicated by three endemic syndromes [14].

a) The yes but syndrome stems from human nature and the customers' inability to experience the software as they might a physical device.
b) Searching for requirements is like searching for Undiscovered Ruin; the more you find, the more you know remain [49].
c) The Customer and the Developer syndrome reflect the profound differences between the two and making communication difficult.

6.1.2 Problems of Requirement Specification

The previous Section 6.1.1 was focused on the process of eliciting customer needs, analyzing the problem, and collecting, and managing the desired product features. The specification process of the requirement engineering process is now discussed. A complete set of requirements can be determined by defining the system inputs, outputs, functions, attributes, and attributes of the system environment [49]. One of the most difficult challenges in the requirement specification process is to make requirements this much detailed enough that can be well understood without explaining the system and defining a whole host of things that may be better to left to others further in the process. The goal of requirement specification is to find the balance point where the investment in the requirement provides the right amount of specificity and leaves the right amount of ambiguity for others to resolve further process.

Few specification techniques are proposed to solve this ambiguity problem. Use Case [25] has achieved a degree of popularity and become a common use for expressing requirements for a system. Using the UML and object-oriented methods the system can be well implemented.

However, the most popular and the best technique for documenting requirements were to use natural language and to simply them write down all in an organized fashion. This technique was revised and improved over the many projects, and eventually a number of standards developed for these documents, including IEEE (Institute of Electrical and Electronics Engineers) 830: Standard for Software Requirement Specification [27]. IEEE Standard 830-1998 also describes the characteristics of a good software requirement specification (eight quality measures) [27]. A software requirement specification should be:

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for Importance and/or Stability
- Verifiable
- Modifiable
- Traceable

If description of the requirement is too complex for a natural language and if specification misunderstanding cannot afforded, then should consider writing that portion of the requirements using a technical methods approach. Some technical requirement specification methods are as follows:

- Pseudocode
- Finite State Machines
- Decision Trees
- Activity Diagrams (Flowcharts)
- Entity Relationship Models
- Object-Oriented Analysis
- Structured Analysis

6.1.3 Problems of Requirement Validation and Verification

Verification and Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it fulfills its intended purpose [22]. Building the right system right depends on continually confirming that the development is on track and that the results are correct, as well as being able to deal with change during development [49].

Verification is a process of requirement engineering whose goal is to assure continually that development of system fully satisfies all the customers' needs. IEEE defines verification as:

The process of evaluating a system or component to determine whether the products of a given phase satisfy the conditions imposed at the start of that phase [28].

Verification is supported by the use of traceability techniques to relate parts of our project to one another [49]. By using traceability, all project elements can be verified.

Validation goal is to demonstrate that the product conforms to the requirements and may gain customer acceptance of the final result. IEEE defines validation as:
The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements [28]. Validation techniques uses to ensure that all project elements are tested properly and all tests have useful purpose. Being able to talk to someone and write and take good and accurate notes and being able to read body language are more important than any tool. Tools make communication, traceability and monitoring of requirements easier but they don't make the requirements any more accurate.
Some issues of validation and verification are as follows:

- Their intensive laboring demands and time consumption [30].
- Their confinements by the definition of the term defined by different standards [51].

6.1.4 Problems of Requirement Analysis

Requirement analysis encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, analyzing, documenting, validating and managing software or system requirements [33]. It includes customers' business context understanding, functions that product must perform, the performance levels that must achieve, and the external systems for which it should be compatible. Techniques used to for this kind of understanding include customer interviews, and use cases of software functionality. Formal requirement specification is the resultant output of the analysis, which serves as input to the next step.

In this way it is supposed to work. But in reality, there are a number of problems with this model, and these may cause delays and invite errors in the further steps of the process. Developers experience some issues during this phase.

- Customers don't really know what they want
- Requirement changes during development of the project
- Customers have unreasonable timelines.
- Communication gaps exist between customers, engineers and project managers [56].

6.2 How Web can Overcome the Requirement Engineering Problems

Web can help in requirement engineering process to overcome the problems in that process. Now for each processes of requirement engineering how web can be helpful to overcome the problem studied above has been discussed.

6.2.1 Use in Requirement Elicitation

Requirement elicitation is a process of collecting requirements from users, customer, and stakeholders, it is a communications-intensive process [40], so by definition, any tools which improve communications will improve requirements and the question arises, how to best utilize the web in requirement elicitation. In this context, how web can be an advantage for requirement engineering has been discussed.

First of all, let discuss what social media which is a part of web2.0 is. Social media refers to the means of interactions among people in which they create, share, and exchange information and ideas in virtual communities and networks [2]. Social media is a web-based technology transforms and broadcasts media speech into social media dialogues. Social media differentiates from traditional media in many aspects such as quality, reach, frequency, usability, immediacy and permanence [1] [34]. Company's internal network can host social media tools which can be used by their employees. Also, this is not necessary that all social media tools need to be publicly broadcasts to anyone. For example, Facebook restricts user content to its friends circle only.

Core principles that underline the web solutions [50] are as follows:

a) Involvement
b) Integrative
c) Uncloudedness
d) Freedom
e) Continuation
f) Appearance

Let's go through each point and discuss about how it can help in requirement elicitation.

Involvement

Participation is crucial for requirements gathering, can get input from the right person in order to gather accurate requirements. Single person's requirements may be of no use unless the product will be used by that person only. Whether throughout the world there are few user representatives or million, web can help to hear what their requirements are.

Integrative

Web can not only help in collecting the requirements but can also help in discussions and disagreements around the world. Web can collect and acquire content, may be files, photos, videos, or anything else. Hence requirement can fit in perfectly because they are a collection themselves [52].

Uncloudedness

Social media communication is generally broadcast to the community. Users can see each other's contributions, can add to them, or dispute them, which is a necessity for requirements gathering [52]. Requirements can be publicized, critiqued, and altered based on the feedback of the user community. Social media can be specializing in enable this type of transparency and interaction.

Freedom

Web can provide independence to the users like anyone can contribute at any time and there is no need of coordination between collaborators and this is good for requirement elicitation technique, because one cannot say when inspiration will strike. Requirements reviews are also not perfect sessions where all of the issues come into the picture. This may happen, that after a day one think of something while driving that may significantly change the requirements, then that person needs to communicate as soon as possible, whether meeting is scheduled or not.

Continuation

Content in social media perseveres and doesn't disappear. This is important for requirements gathering, because many times the important discussed point for decisions is forgotten. It would be so valuable to save all the decisions, and also all the discussions happen that take into making that decision.

Appearance

Web can help in emerging ideas that otherwise may remained hidden. This can encourages the person to give their opinion and suggest something that wouldn't have said without the use of web. That opinion can give a topic for discussion that can change the project direction.

It would be better for the product if all relevant problems come into picture during requirement gathering rather than in beta testing. All these factors point to the great usefulness of social media in requirements gathering [52].

6.2.2 Use in Requirement Analysis

Web can integrate documents, analyses and facilitates organizations to act on extracting intelligently from internet conversations on consumer or professional media sites. It can also facilitate to credit internet conversations to specific parts of business, allowing fast responses to marketplace shifts.

On the Basis of business challenges and enterprise goals, web can provide a smart implementation that can be hosted and managed on internet.

- *Analyze Conversation Data:* Web can intensify the current market researches by continuous online monitoring and social conversation data to gathering important topics and content categories and also how those topics and categories are relevant to consumers in the context of their internet community.
- *Identify Opportunities and Threats:* By analyzing professionally or consumer generated media such as articles and blogs, the solution serves as an early-warning system to identify good or bad influences.
- *Quantify Interaction:* By analyzing internet social media sources, one can understand how to reach consumers through improved target and planning.
- *Social Customer Relationship Management Strategy:* By merging customer data gathered from customer generated media such as surveys or web forms and market data from blogs, market research professionals can determine a consumer needs.
- Web provides multi-language support because a bad customer review is valid in Chinese or French as it is in English.
- Web can allow organizations to advance beyond ad hoc social media interactions by maintaining an archive of conversations necessary for ongoing analysis [55].

CHAPTER 7

ANALYSIS OF REQUIREMENT ENGINEERING APPROACHES FOR WEB METHODOLOGIES

ANALYSIS OF REQUIREMENT ENGINEERING APPROACHES FOR WEB METHODOLOGIES

In this chapter, the most common and relevant methods in Web engineering are being discussed, paying special attention in how web application requirements can be specified. In this context, analysis of web engineering methods that introduce a requirements phase in its development process has been done. For each of these method following information has been studied:

- A general description of the method.
- The phases of its development process
- A study of the requirement specification techniques that are proposed by the method. To present a representative example of the requirement specifications obtained by these techniques used them to specify the requirements of a web application for some e-commerce application.
- The main limitations of these techniques.
- Tool support, which focuses on analyzing whether or not tools for supporting the specification of Web application requirements are provided.
- The guidelines and/or tools that are provided to obtain the Web application conceptual model from requirement specifications.

7.1 OOHDM: Object Oriented Hypermedia Design Model

OOHDM was developed by Schwabe and Rossi in 1994 [42]. It was one of the first approaches in providing a methodological solution for the development of Web applications. This approach takes some ideas proposed in HDM [21] and applies them in a well-defined development process based on the Object-Oriented paradigm. OOHDM emphasizes the separation of the navigational aspect from other aspect such as the conceptual aspect and the interface aspect. Other approaches have been further inspired by this idea of separation of aspects. Finally, it is worth to remark that OOHDM is not a closed approach and it is continuously being extended and improved.

Development Process: The development process of this approach is divided into five main phases:

- *Requirements Gathering*: In this phase, the users that must interact with the Web application are identified as well as the users' needs that the web application must support. Early versions of

OOHDM do not consider the requirement phase. This phase was included after defining some extensions to the method.

- *Conceptual Design*: This phase consists in the definition of a conceptual schema where the static aspect of the system is described.
- *Navigational Design*: In this phase, a navigation class diagram and a navigation structure diagram has to be define. 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.
- *Abstract Interface Design*: This phase consists in the description of the user interface in an abstract way. To do this, an abstract interface model is developed using a special technique named ADVs [19].
- *Implementation*: In this phase, the Web application is implemented. It is based on the previous models.

Requirement Specification Techniques: Web application requirements are handled in the phase of Requirements Gathering. This phase is divided into five steps:

1. Identification of Roles
2. Specification of Scenarios
3. Specification of Use Cases
4. Specification of User Interaction Diagrams
5. Validation of Use Cases and User Interaction Diagrams

In this context, requirements are specified as follow: (1) Once user roles are identified, (2) users describe the work that each role must perform by means of scenarios; next, (3) use cases are defined from scenarios and finally (4) use cases are refined with user interaction diagrams (UIDs). For each use case, a UID is defined.

Each UID graphically describes the interaction between the users and the system without considering specific aspects of the user interface. The process to get an UID from a use case is described very carefully in the approach. Figure 7.1, 7.2 and 7.3 shows partial requirement specification of the e-commerce example that has been defined by using the OOHDM approach.

Figure 7.1 shows three scenarios described by different users: Browsing books, Finding a book and Adding a book to the shopping cart. These three scenarios describe a similar work. Then, a use case that supports this work is derived from them shown in Figure 7.2. The UID (shown in Figure 7.3) associated to this use case indicates that: the system provides users first with a list of books category names. From this list, users can select one and then they obtain a list of books. The title, the price and the author name is given for each book. From the list of novel categories, users can also introduce the name of an author. From this name, a list of authors is provided. If users select one author they obtain a list with its books. From the list of books, users can select one and then obtain a description of it. In this description, the title, the price, the edition, the cover, some comments and the author name of the book are provided. Users can also add the book to the shopping cart at this point.

Scenarios

Sc1 – Browsing books
I am a regular user of e-commerce applications. I like starting to browse books from a list of books category such as novel, magazine, study, engineering, etc. Then, I like obtaining a list of books from which I can select the desired one.

Sc2 – Finding a book
I like novels very much. However I only buy the books of my favourite author. Then, I like those e-commerce applications that allow me to quickly access to the books of a specific author. Then a search by author is mechanism that should be supported.

Sc3 – Adding a book to the shopping cart
I am the manager of the marketing department. We need that user can early add the book to the shopping cart when they find it. Furthermore, we need to automatically update the stock after adding the book.

Figure 7.1 Example of Requirements Scenario in OOHDM

Use Cases

Use Case: Adding a book to the shopping Cart
Scenarios: Sc1, Sc2, Sc3
Roles: Anonymous users, Registered Clients
Description:

1. The application provides the user with a list of books category.
2. The user selects a category.
3. The application returns the list of books that belongs to the selected book category.
4. The user can select a book of perform a search of book by author.
5. If the user select a book returns a detailed description of the selected book.

Figure 7.2 Example of Use Case in OOHDM

UIDs

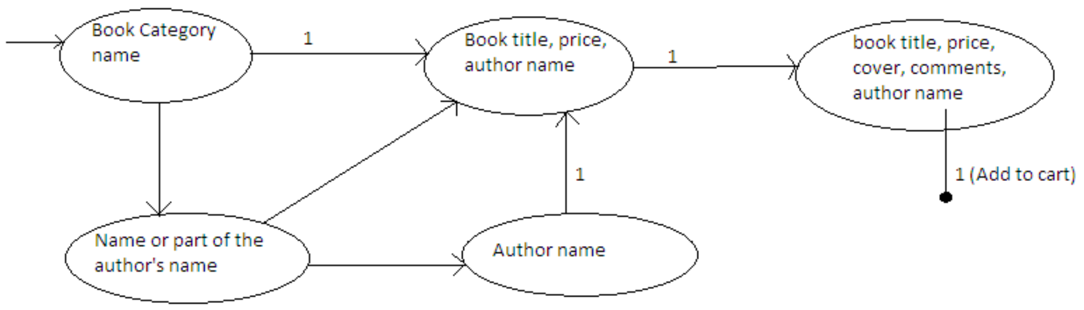

Book Category name
1
Book title, price, author name
1
book title, price, cover, comments, author name
1 (Add to cart)
1
Name or part of the author's name
Author name

Figure 7.3 Example of Requirement Specification in OOHDM

Requirement Specification Limitations

- Navigation is captured for each use case individually from its associated UID and then navigation is not captured from a global view of the system.
- Navigational requirements captured by UID are not properly related to each other e.g. how users access different navigational structures captured by different UIDs.
- Transactional requirements can be textually described in a use case they cannot be completely specified in UIDs.
- A sequence of system operations cannot be defined in a UID.
- Data requirements must be extracted from interactions between the user and the system which is not always adequate. For instance, data that is only required to support internal operations of the system is not properly considered.

Tool Support: Currently, there is no tool which supports the Requirements Gathering phase. Then, each step must be manually performed.

Guidelines for Obtaining Conceptual Models: This approach presents guidelines in order to help analysts to define the conceptual and navigational schema from requirement specifications [42]. These guidelines are defined from a set of rules indicate how the elements that define both a class diagram (conceptual schema) and a context diagram (navigational schema) can be systematically derived from UIDs.

These rules are textually described by using natural language. No formalism is used in the rule definition. In this context, certain ambiguity degree is introduced in the rule definition (derived from the use of natural language). This aspect becomes critical if someone considers that rules must be manually interpreted and applied. No tool is provided to support the application of these rules.

7.2 WSDM: Web Site Design Method

WSDM was developed by De Troyer and Leune in 1998 [13]. It is a user-centered approach. WSDM defines a Web application by describing the requirements of the different groups of users that must interact with it. It was one of the first approaches in considering the problem of the diversity of users in Web applications. This approach is continuously being extended and improved.

Development Process: The development process of this approach is divided into five main phases:

- *Mission Statement Specification*: In this phase, the purpose and the subject of the web application must be expressed. Declaration of the target audience is also must.
- *User Modeling*: In this phase, users are classified and grouped in order to study system requirements according to each user group.
- *Conceptual Design*: In this phase, both 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*: This phase consists in the translation of the models defined in the conceptual design phase into an abstract language easily to be understood by the computer.
- *Implementation*: In this phase, the implementation design result is written in a specific programming language.

Requirement Specification Techniques: Requirements are handled in the Mission Statement Specification and User Modeling phases. Taking as source the description of the mission statement, the User Modeling phase is performed in two steps:

(a) Audience Classification and (b) Audience Class Characterization, in these two steps the concept of class audience plays a key role. A Class Audience is a group of potential visitors that has the same functional requirements [13].

1. In the Audience Classification step, people are classified according to the activities related to the purpose and subject of the Web application. This formally identifies the different Audience Classes. 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.
2. In the Audience Class Characterization step, Audience Classes are analyzed in detail. To do this, textual templates with the help of a data dictionary are used to define the functional requirements for each group of users. Figure 7.4, 7.5 and 7.6 shows a partial requirement specification of the e-commerce example that has been defined by using the WSDM approach. Figure 7.4 shows the mission statement. In Figure 7.5, a partial result of the audience classification step has been shown. Three audience classes shown in Figure 7.6 have been identified: Customers, Visitors and Administrators. Customers can query product information and purchase products; Visitors can only query product information; Administrators can manage product information. Finally, in Figure 7.6 a partial characterization of the audience class Visitors has been shown.

Mission Statement

To support users (Visitors, customers, and Admin) in:

1. The purchase of books and games
2. The management of the system throughout the internet.

Figure 7.4 Example of Mission Statement in WSDM

Audience Classification

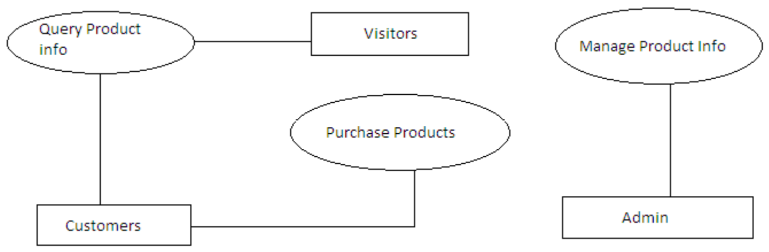


Figure 7.5 Example of Audience Classification in WSDM

Audience Characterization

Visitor

1. View the different type of products. For each type view the list of available products.
2. View product descriptions. For each product, get a description. Link the product's name in the list of products to product description web page.
3. Search products. For every type, users can search products by introducing a specific criterion.

Figure 7.6 Example of Audience Characterization in WSDM

Requirement Specification Limitations

- The main drawback of this approach is that requirements are basically defined textually.
- Describing navigational requirements by textual descriptions which may need to consider multiples possibilities of navigation information is not always an easy task.
- Textual descriptions may be overloaded and difficult to manage in the development of complex Web applications.
- Works such as states, textual specifications are hard to maintain when are used in a validation process with users, mainly because of the lack of precision and conciseness.

Tool Support: Currently, there is no tool for supporting the requirement phases. Then, requirements must be manually specified.

Guidelines for Obtaining Conceptual Models: There are no published works that present guidelines that help analysts to define the Web application conceptual model from the requirement specification.

7.3 SOHDM: Scenario-Based Object-Oriented Hypermedia Design Methodology

SOHDM was developed by Heeseok Lee, Choongseok Lee, and Cheonsoo Yoo in 1998 [26]. This approach considered RE activities in the development process of a Web application from its initial version. This approach takes some ideas from OOHDM [42]. However, as its name indicates it is mainly based on the use of scenarios.

Development Process: The development process of this approach is divided into six main phases:

- *Analysis*: In this phase, the requirements of the web application are describe by using scenarios.
- *Object model realization*: This phase consists in creating a class diagram where the static structure of the system is defined.
- *View Design*: In this phase, explanation of how the system will be presented to the user has to be expressed.
- *Navigational Design*: In this phase, a navigational class model is developed in order to express the possibilities of navigation in the system.
- *Realization of Implementation*: This phase consists in designing the Web pages and the flow among them, other detailed interface aspects and a relational database.
- *Construction of System*: In this phase the Web application is finally built.

Requirement Specification Techniques: Requirements are handled in the Analysis phase. SOHDM proposes to specify web application requirements through three main steps:

1. Definition of the scope of the system, which delimits the Web application to be developed from external entities. In order to define the scope of the system SOHDM proposes the System Scope diagram that is based on the well-known Data Flow Diagrams (DFD) [15], although context diagrams (e.g., zero level DFD) are a good alternative. In the System Scope diagram, identification of external entities as well as the information exchanged between these external entities and the system has to be done.
2. Identification of events that trigger the communication between external entities and the application. For each external entity one or more events are identified. Events are defined in a specific table.
3. Association of scenarios to the identified events. Scenarios are described by means of Scenario Activity Charts (SACs) that is a notation created by the authors. SAC describes business processes according to actors. An actor is an operator of specific activities, i.e., a creator of events. Figure 7.7, 7.8 and 7.9 shows partial requirement specification of the e-commerce example that has been defined by using the SOHDM approach. Figure 7.7 shows the scope of the Web application where three external entities are identified: Visitor, Customer and Administrator. Figure 7.8 shows the events identified for each external entity. In Figure 7.9, the SAC that describes the event Query Product Information is shown. External entities are the primary candidates for actors in SACs. An event is a starting point, a trigger of a scenario. An activity is an operation by which an actor completes a scenario. A decision node represents an

activity that enables for an actor to choose the next activity. An activity flow is a sequence of activities. Termination is the end of a scenario.

Requirement Specification Limitations

- The main drawback of this approach is that SACs are mainly focused on describing transactional requirements where the exchange of information between the system and the user is almost limited to required operation parameters.
- Description of navigational and data requirements is not always easy in the case of Web applications that are mainly focused on just providing information.
- The major part of these Web applications allows users to navigate information (data and navigational requirements) without performing any operation (transactional requirements).
- To represent all the different possibilities for navigating information with a DFD-based notation can overload the requirement specification, making them difficult to manage and understand.
- This technique does not provide explicit support for the specification of data requirements. Only the data that is requested by the system in order to perform an activity can be defined in a SAC.

Tool Support: Currently, there is no tool for supporting the Analysis phase. Then, every diagram must be manually defined.

Guidelines for Obtaining Conceptual Models: There are no published works that present guidelines that help analysts to define the Web application conceptual model from the requirement specification.

System scope

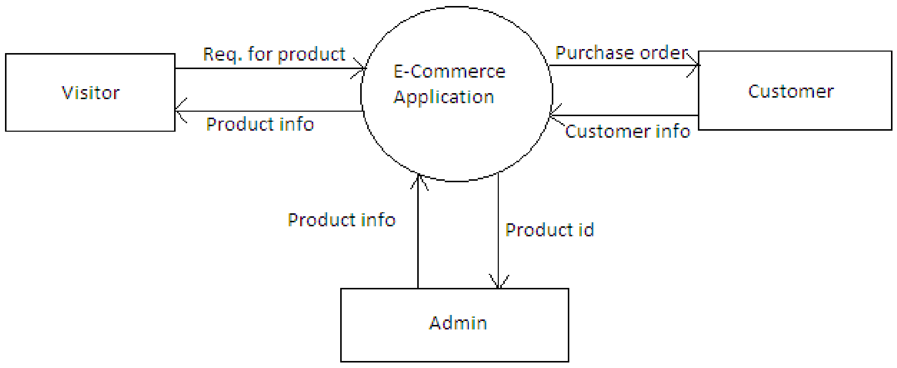


Figure 7.7 Example of System Scope in SOHDM

Events

Source Entity	Event Name
Visitor	Query Product Info
Customer	Purchase Products Query Product info
Admin	Manage Product info

Figure 7.8 Example of Event in SOHDM

SAC

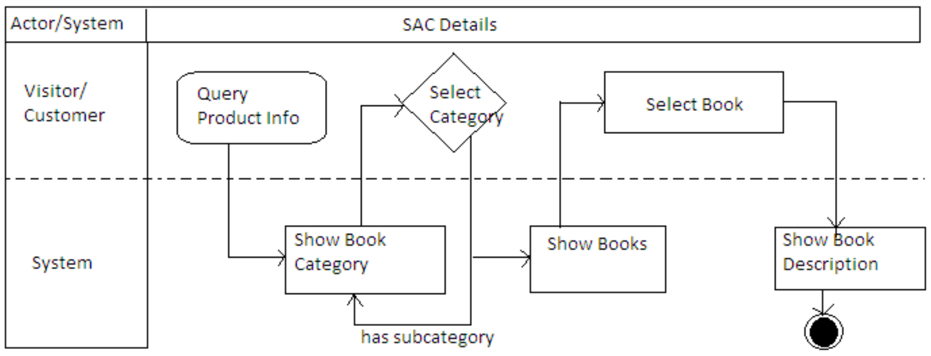


Figure 7.9 Example of Requirement Specification in SOHDM

7.4 UWE: UML-Based Web Engineering

UWE has been first introduced by Nora Koch in 1998 and after that many modification has been done by Nora Koch [35]. UWE is a methodological approach for the development of Web applications that completely bases its diagrammatic techniques on UML. This approach is currently open and a lot of research work is continuously being presented in order to improve it. UWE proposes a semi-automatic design process supported by the case tool ArgoUWE6.

Development Process: The development process of UWE is based on the Model Driven Architecture [MDA]. The aim of such a MDD process is the automatic model transformation in each step based on rules defined at meta-model level. This process has three main stages:

- The process starts with the specification of requirements. These requirements define the CIM model.
- Platform independent analysis models (PIMs) are derived from the requirements (often based on additional information). On the PIM level, the separate concerns of Web applications (the content, the navigation, the business processes, and the presentation) are designed in separate models. These models are integrated into a so-called big picture which merges all the concerns together and is used in validation of the design models.
- Finally, the platform specific models (PSMs) are derived from this validation model, from which program code can be generated.

Requirement Specification Techniques: Initial versions of UWE propose the specification of requirements by means of use case diagrams together with textual descriptions of use cases. In recent works [35], UWE has been extended in order to support the UML profile for Web Requirements Engineering (WebRE). This profile has been defined by the UWE's authors in collaboration with the authors of the Web engineering method NDT. Main focus is on the most recent approach based on WebRE, considering that the other works could be obsolete.

In this context, UWE currently proposes the use of use cases diagrams and activity diagrams in order to capture web application requirements. Use case diagrams are used to represent an overview of the functional requirements while activity diagrams provide a more detailed view. UWE distinguishes between two types of use cases:

Navigation use cases which comprise a set of browse activities that the user will perform to reach a target node. A browse activity is the action of following a link. A browse activity can be enriched by search actions. A search action has a set of parameters, which let define queries on content. The results are shown in the target node [35].

The second kind of use case is the *Web Process*, which is refined by activities of type browse, search, and at least one user transaction [35]. A user transaction represents more complex activities that are expressed in terms of transactions that have been initiated by the user, like checkout in an e-shop or an online reservation.

Figure 7.10 and 7.11 shows a partial requirement specification of the e-commerce example that has been defined by using the UWE approach. Figure 7.10 shows the use case diagram. Figure 7.11 shows the activity diagram that describes the navigational use case Find book. According to this diagram, when users activate the link List book Category they access the node book Category. From this list users can select one category and then access the node book. Another way of accessing this node is by searching authors from the node book Category.

Requirement Specification Limitations

- This approach considers only those requirements which support each use case individually.
- UWE does not analyze navigational requirements from a global vision of the system.
- Data requirements must be extracted either from the parameters that are needed for performing searches or from the information that is modified by transactions performed by the user [35].
- This aspect makes it difficult to consider the requirements related to data that is only required to support internal operations of the system.
- Activity diagrams in UWE allow indicating which type of information is accessed by users by navigating nodes (e.g. books, Novel Categories, etc.). However, it is not clear how details about the information provided in each node can be given at the requirements level.

Tool Support: UWE is supported by the ArgoUWE tool. This tool is a plugin defined over the open source modeling tool ArgoUML9. ArgoUWE is focused on supporting the modeling activities of the UWE development process as well as activities related to the semi-automatic generation of code from models. In this context, both use case diagrams and activity diagrams can be graphically defined by means of ArgoUWE. Additional information about ArgoUWE can be found in its Web page10.

Guidelines for Obtaining Conceptual Models: A model-to-model transformation is proposed in order to obtain draft versions of both UWE navigational models [35] and UWE content models from requirement specifications. These model-to-model transformations are based on the definition of mappings between the WebRE meta-model and the UWE meta-model. These mappings are defined as QVT transformations (QVT). However, authors do not provide a tool for automatically applying these transformations. They are currently looking forward to tools supporting the QVT transformation language. In the meantime, they are gathering experience with other transformation languages and techniques, such as ATL and graph transformations.

Use Cases

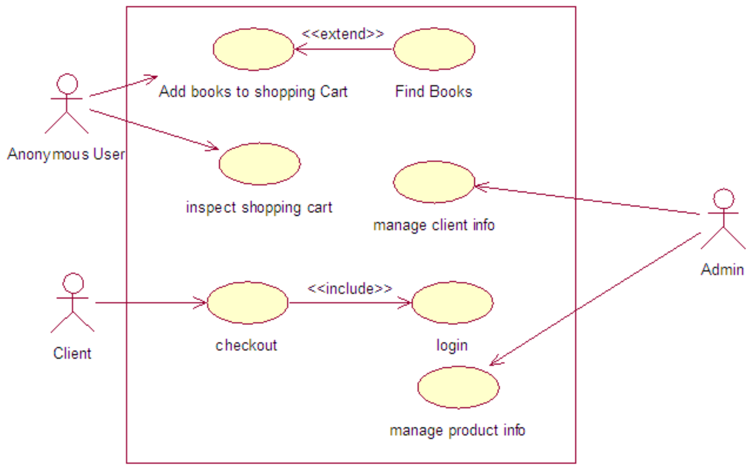


Figure 7.10 Use Case in UWE

Activity Diagram

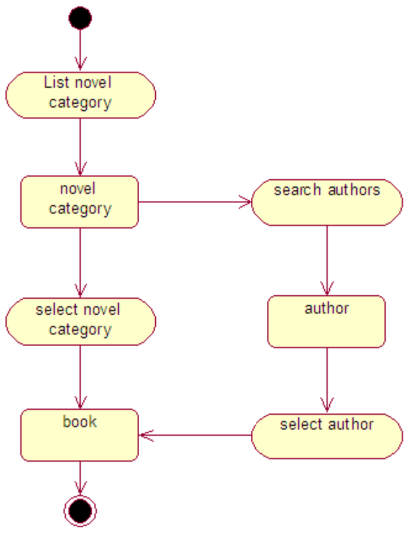


Figure 7.11 Example of Activity Diagram in UWE

7.5 Building Web Applications with UML

This approach is an extension to UML introduced by Jim Conallen in 1999 [31]. It is based on the Unified Process [29]. The graphical notation of UML is extended throughout stereotypes. These stereotypes can be freely obtained and incorporated to the IBM tool Rational Rose.

Development Process: The development process of this approach is divided into eight main phases:

- *Planning*: This phase consists in the analysis of the organization needs and the definition of a plan of how the rest of the phases in the development process must be performed.
- *Requirements*: In this phase, the requirements of the system are gathered and defined by means of UML-based models such as use case diagrams or activity diagrams.
- *Analysis - Design:* These two phases consist in the transformation of requirements into a design that can be realized in software. These two phases can be done separately or combined as a part of one set of activities.
  - i) Analysis consists in obtaining an analysis model from requirements. This model is made up of classes and collaboration of classes that represent the dynamic behavior of the system. The emphasis of this phase is on ensuring that all the functional requirements are supported.
  - ii) Design consists in the use of non-functional requirements and architecture constraints in order to refine the analysis model into something that can be coded.
- *Implementation*: In this phase, the Web application design is mapped into code and components. A set of mappings are provided to facilitate this. Next, code must be built into binaries.
- *Test*: In this phase, each member of the development team must test its own work. The goal of this phase is to have every delivered artifact, or component, tested before any other member of the team uses it.
- *Evaluation*: The whole system is evaluated to ensure that correctly supports users' needs.
- *Deployment and Maintenance:* In this phase, the system is delivered and installed. After that, maintenance activities must be performed.

Requirement Specification Techniques: Requirements are handled in the Requirement phase. Although this approach proposes multiple new UML stereotypes for supporting the Web application development they are all mainly focused on the analysis and design phase. The technique proposed to specify requirements is based on the use of traditional use cases together with activity diagrams and/or sequence diagrams. Activity diagrams are proposed to describe use cases. Sequence diagrams are recommended to be defined in order to better clarify which is the basic flow of the use case and which are the alternative flows. Figure 7.12 and 7.13 shows a partial view of the Conallen's requirements specification for the e-commerce example. Figure 7.12 shows the use case diagram. Figure 7.13 shows the activity diagram associated to the use case find book.

Use Case Diagram

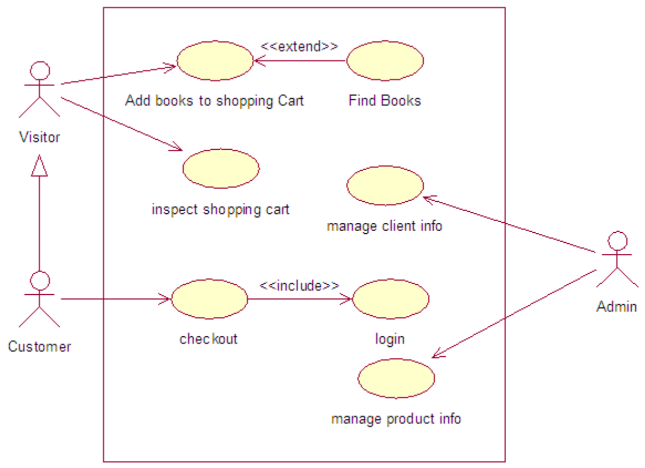


Figure 7.12 Example of Use Case in Conallens' Approach

Activity Diagram

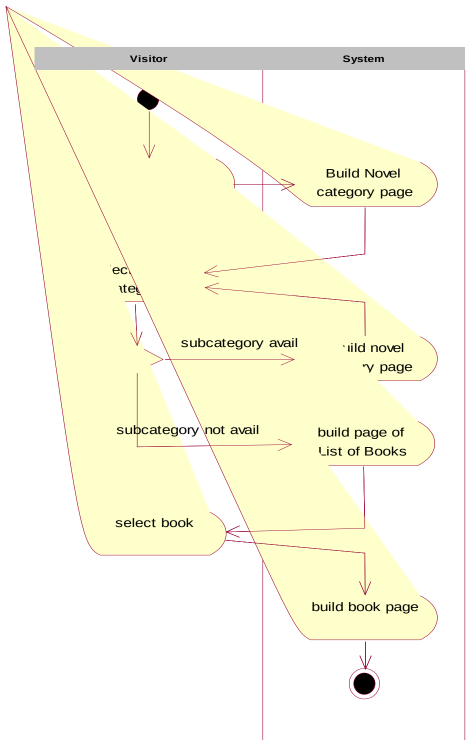


Figure 7.13 Example of Activity Diagram in Conallens' Approach

Requirement Specification Limitations

- This approach does not provide specific techniques for handling web application requirements and it proposes the use of traditional use cases.

- Use cases become sometimes insufficient and ambiguous when they are used to capture the navigational aspect of Web applications.
- This approach is a diagrammatic technique such as Activity Diagrams is proposed to complement use cases [31]; it is based on concepts that are close to the implementation level.
- The technique proposed by Conallen is mainly focused on capturing the behavior of both the user and the system. Few support for detailing structural aspects such as the data that the system must store are provided.

Tool Support: Since this approach is fully based on UML it is supported by general modelling tools such as Rational Rose. In fact, profiles proposed by this approach are available to be incorporated in this tool.

Guidelines for Obtaining Conceptual Models: There are no published works that present guidelines to facilitate the creation of web application conceptual models from requirement specifications.

7.6 WebML: Web Modeling Language

WebML was developed by Piero Fraternalli and Stephano Ceri in 2000 [39]. WebML is a high level modeling language for Web applications. WebML follows the style of both Entity-Relationship and UML offering a proprietary notation and a graphical representation using the UML syntax. This approach is currently not closed and it is continuously being extended and improved. Finally, WebML is one of the few approaches that present a tool that supports its development process. This tool is called WebRatio14 and it is being currently applied in an industrial environment.

Development Process: The development process of this approach is divided into six incremental main phases:

- *Requirement Analysis*: This phase focuses on collecting information about the application domain and the expected functions, and specifying them.
- *Conceptual Modeling*: This phase consists in defining WebML-based conceptual schemas, which express the organization of the application at a high level of abstraction, independently from implementation details.
- *Implementation*: This phase consists in the translation of the WebML-based conceptual schema into a specific implementation technology.

- *Testing and Evaluation*: In this phase, the Web application is tested and validated in order to improve its internal and external quality.
- *Deployment*: This phase consists in deploying the Web application on the top of a specific architecture.
- *Maintenance and Evolution*: In this phase, the application is maintained and possibly evolved after it is deployed.

Furthermore WebML also promotes a strategy of rapid prototyping from the conceptual schema in order to early detect misunderstanding in the captured requirements [39].

Requirement Specification Techniques: Although WebML considers RE activities in its development process it does not prescribe any specific format for requirement specification. According to its authors: *Requirement specification is the activity in which the application analyst collects and formalizes the essential information about the application domain and expected functions, this aspect do not significantly differ from requirement collection for traditional applications* [39]. Although no specific techniques for web application requirement specification are given, the different published works that are related to WebML how authors suggest some tabular formats for capturing types of users or propose the use of UML uses cases and activity diagrams in order to specifying functional requirements can be seen [39]. Furthermore, the use of mock-ups (sketches) is also suggested for the specification of the site view. Figure 7.14, 7.15 and 7.16 shows a partial view of the requirement specification of the e-commerce example that has been defined by using the techniques proposed by WebML. In particular, Figure 7.14 shows the description of the Visitor user type. In Figure 7.15, an activity diagram that group in phases the different activities that can be performed by the identified types of user has been shown. In Figure 7.16, a use case diagram that provides a more detailed description of one of the phases included in the activity diagram of Collect Products has been shown.

Requirement Specification Limitations

- The main drawback of this approach is that it is focused on the use of traditional techniques such as use cases without an explicit extension for supporting web application requirements.
- Navigational requirements are many times not supported properly by use cases.
- Other works such as England & Finney, Burdman, Overmyer and Lowe also state that new RE methods are needed for Web applications because of RE methods for traditional software are not

appropriate to support distinctive characteristics of the Web application development such as, for instance, navigation.

Tool Support: As introduced above, WebML is supported by the commercial tool WebRatio. However, this tool is mainly focused on (1) providing support for the Conceptual Modeling phase and (2) providing mechanisms for the automatic generation of code. The specification of requirements is not supported by this tool (at least in its current version 5.0).

Guidelines for Obtaining Conceptual Models: There are no published works that present guidelines to define the Web application conceptual model from the requirement specification.

User Type Description

Visitor

Description: It represents all the users that can access the public part of the web application.
Profile data: No profile required
Object access in read mode: Products
Object access in write mode: Shopping cart
Relevant usage scenarios: Query Product info, add product to shopping cart

Figure 7.14 Example of Use Type Description in WebML

Activity Diagram

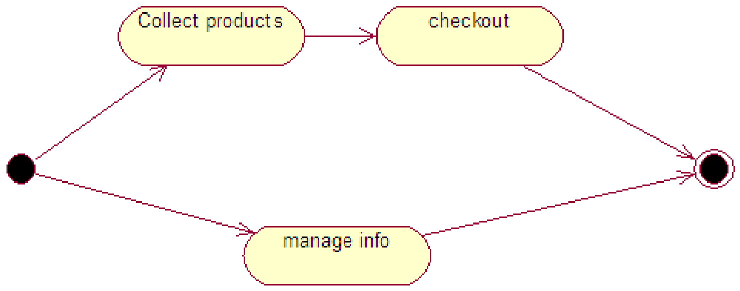


Figure 7.15 Example of Activity Diagram in WebML

Use Case Diagram

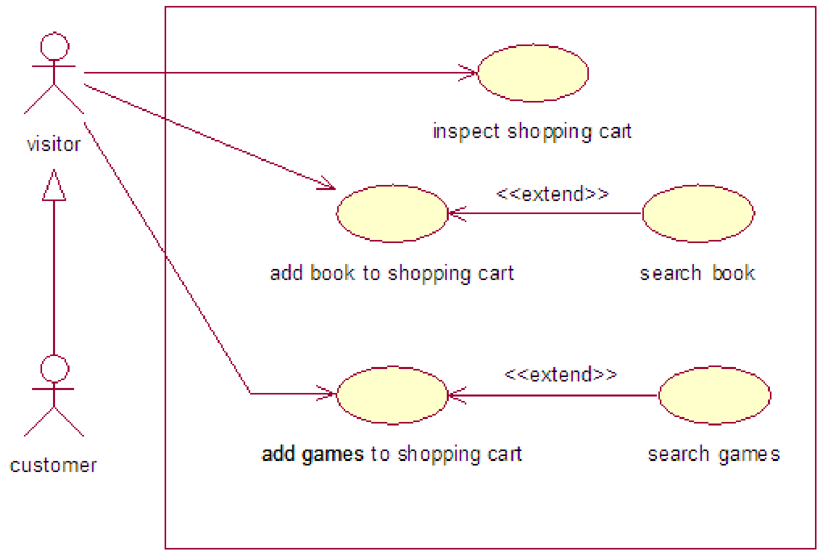


Figure 7.16 Example of Use Case Diagram in WebML

7.7 OO-H: Object-Oriented Hypermedia Method

Object-Oriented Hypermedia Method was presented by Jaime Gomez, Cristina Cachero and Oscar Pastor in 2000[23]. This approach emerged formerly as an extension of OO-Method [23], a design method for object-oriented systems. However, in the most recent publications authors dissociate OOH from OO-Method and focus on dealing with personalization of web sites [20]. This approach is supported by the tool Visual Wade.

Development process: The development process of this approach is divided into four main stages:

- *Requirement Analysis*: In this phase, the web application requirements are specified for each different type of user.
- *Engineering*: This phase involves all the activities related to the analysis and design of the software product. In particular, these activities are five: two analysis activities (domain and navigational analysis) and three design activities (domain, navigation and presentation design).
- *Construction and Adaptation*: This phase constitutes the implementation of the Web application.
- *Client Evaluation*: In this phase, clients evaluate the developed Web application.

Requirement Specification Techniques: Requirements are handled in the Requirement Analysis phase. However, OO-H is mainly focused on both providing abstract mechanisms in order to capture Web applications at the conceptual level and defining strategies for the automatic generation of code [23]. RE activities for Web applications are considered by paying less attention. OO-H proposes only the use of use case diagrams.

In an extension of the method [35], OO-H proposes to complement use cases with activity diagrams. However, these activity diagrams are used in the analysis activities that are included in the Engineering phase. They are not used as technique for the specification of web application requirements. Figure 7.17 a partial view of the use case diagram proposed by OO-H in order to specify the requirements of the e-commerce example.

Requirement Specification Limitations

- The main drawback of this approach is the recommendation of traditional techniques such as use cases for the specification of web application requirements.

Tool Support: OO-H is supported by the tool Visual Wade. This tool provides a complete graphical support for performing the domain and navigational analysis. However, this tool does not provide support for creating requirement specifications.

Guidelines for Obtaining Conceptual Models: There are no published works that present guidelines to define the Web application conceptual model from the requirement specification.

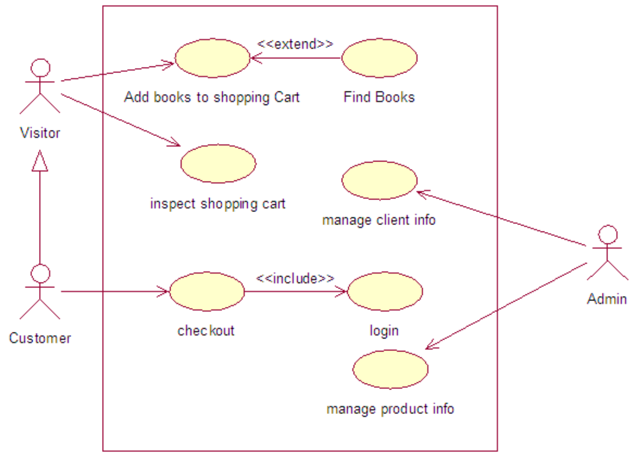


Figure 7.17 Example of Requirement Specification in OOH

7.8 W2000

W2000 was developed by Luciano Baresi, Franca Garzotto, and Paolo Paolini in 2001 [7]. This method is presented as an extension and customization of UML with web design concepts borrowed from the Hypermedia Design Model (HDM) [7].

Development Process: The development process of this approach is divided into five main phases:

- *Requirement Analysis*: This phase consists in the study of the web application requirements by means of UML use cases.
- *State Evolution Design*. In this phase, analysts must indicate how contents evolve. This phase is not mandatory, but it is required only for applications with complex behaviors.
- *Hypermedia Design*. This phase consists in the design of the Web application navigational structure. This phase is based on the use of HDM.
- *Functional Design*. This phase specifies the main user operations of the application.
- *Visibility Design*. In this phase, different perspectives of the Web application are defined for each different type of user.

Requirement Specification Techniques: Requirements are handled in the Requirement Analysis phase. The requirement analysis proposed by W2000 is performed from two main activities: functional requirement analysis and navigational requirement analysis [7]. After identifying the different types of user that can interact with the Web application, the functional requirement analysis identifies the main user operations while the navigational requirement analysis indicates the main information and navigation structures needed by the different users.

In order to perform both activities W2000 proposes the use of UML use case diagrams. Two types of use cases are proposed: functional use cases and navigational use cases [7]. The functional use case diagram is a typical use case diagram that identifies the main functionalities and associates them with types of user. The navigational use case diagram indicates the navigation capabilities associated to each type of user.

These navigational capabilities can be constrained with accessibility conditions. Figure 7.18 and 7.19 shows a partial version of the W2000 use case diagrams for the e-commerce example. Figure 7.18 shows a functional use case diagram that identifies functionalities such as login or management of product information. These functionalities are associated to the Customer and Administrator types of user, respectively. Figure 7.19 shows a navigational use case diagram that identifies navigational capabilities such as browse books or inspects shopping cart items. Notice how this last capability presents an accessibility constraint. Visitors only can inspect the items that they have added to the shopping cart.

Functional Use Case Diagram

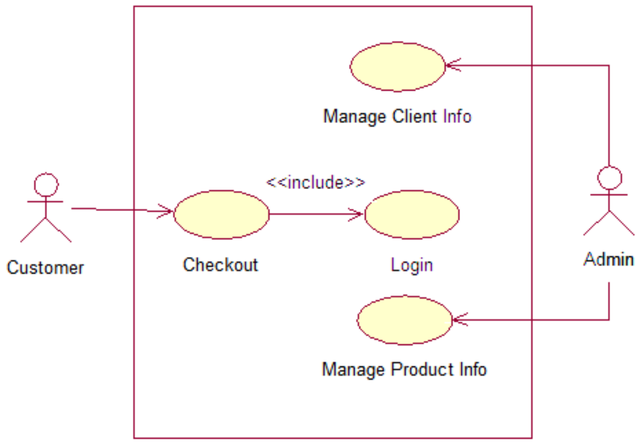


Figure 7.18 Example of Functional Use Case Diagram in W2000

Navigational Use Case Diagram

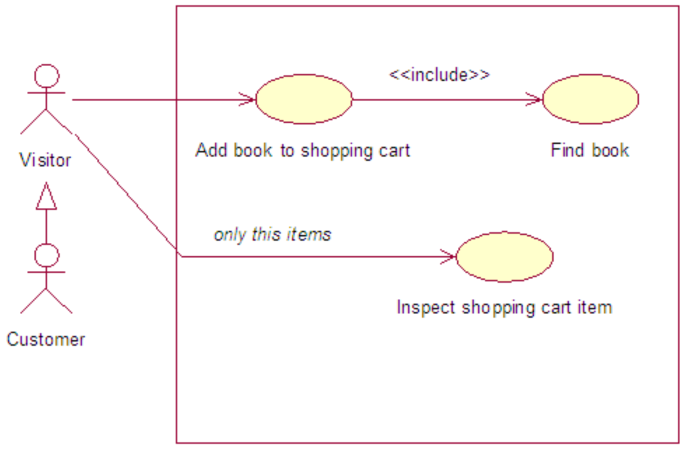


Figure 7.19 Example of Navigational Use Case Diagram in W2000

Requirement Specification Limitations

- W2000 proposes a special use case diagram for capturing navigational requirements; it only allows identifying navigational capabilities.
- Techniques for describing at the requirements level how these capabilities must be supported by the Web application (such as the UIDs of OOHDM or the activity diagrams of UWE) are not provided.

Tool Support: Currently, there is no tool which supports the RE activities in this approach.

Guidelines for Obtaining Conceptual Models: There are no published works that present guidelines to facilitate the creation of Web application conceptual models.

7.9 NDT: Navigational Development Techniques

NDT was presented by Maria Jose Escalona in 2004 [18]. NDT is a methodological process for the Web application development that it is focused on the requirements and analysis phases. It proposes the intensive use of textual templates in the requirement phase and the systematic derivation of analysis models from these templates. This approach proposes the use of prototypes to validate requirements. The NDT tool [18] has been developed in order to support this approach.

Development process: The development process of this approach is divided in three main stages:

- *Requirement Treatment*: In this phase, the Web application requirements are collected and described.
- *Analysis*: In this phase, analysis models are systematically derived from the requirement specification. These analysis models are the conceptual model and the navigational model.
- *Prototyping*: This phase consists in the development of Web application prototypes from analysis models. These prototypes are used to validate requirements.

It is worth to remark that the main goal of NDT is to obtain three results: (1) the requirements catalogue, (2) the requirement analysis document and (3) the web application prototype. These results must be taken as source for the design and implementation phases, which are not handled by NDT.

Requirement Specification Techniques: NDT is an approach fully developed to handle web applications requirements. Thus, this is one of the approaches that provide a more complete support for the specification of web application requirements. This approach also fits (together with UWE) the UML profile for web requirements [18]. In order to specify requirements, NDT basically proposes the use of uses cases diagrams and formatted templates. NDT classifies requirements into the following types:

storage information, actor, functional, interaction and non-functional requirements. For each type, NDT defines a special template, i.e. a table with specific textual fields that are completed by the development team during the phase of requirement elicitation.

Figure 7.20 and 7.21 shows partial requirement specification of the e-commerce example that has been defined by using the NDT approach. Figure 7.20 shows the use case diagram. Figure 7.21 shows an example of template for information storage requirements. In particular, this template defines the use case *login*. The other types of requirements such as storage information, actor, or interaction are described by using similar templates.

Use Case Diagram

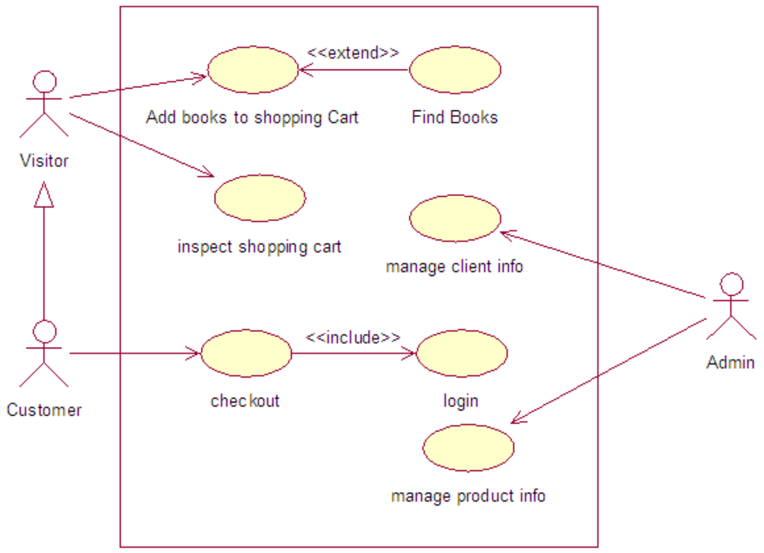


Figure 7.20 Example of Use Case Diagram in NDT

Formatted Templates

Req-01	Login	
Description	Authentication to allow access to the checkout process	
Actors	Use Case Actor	
	AC-01. Web User	
Normal Sequence	**Step**	**Action**
	1	The system asks for the user id and password and option to

		remember both user id and password
	2	The user puts user id and the password
	3	The user id and password are checked
	4	The user id and password is stored if the field remember is true
	5	Access to checkout is allowed
Exceptions	**Step**	**Action**
	4.1	The user is not registered, so the user executes Req-02
	4.2	The user id or/and password not valid, continue to step 1.

Figure 7.21 Example of Functional Template in NDT

Requirement Specification Limitations

- In the development of complex Web applications there may be a lot of different possibilities of navigation. To specify all of them by using just textual templates is not always easy.
- The use of textual templates instead of other diagrammatic tools such as the UIDs of OOHDM or the activity diagrams of UWE makes it difficult to obtain from the requirement specification a complete vision a big picture of the Web application navigational aspect.
- In complex Web applications, the amount of textual templates that must be defined may result very difficult to manage and maintain, and as the own authors of NDT states [18], templates are not easy to complete as they require intensive interviews.

Tool Support for RE activities: NDT is supported by the NDT tool [18]. This tool provides fully support to the whole development process of NDT: it provides support for the specification of requirements, the analysis of them and the generation of HTML-based prototypes.

Guidelines for Obtaining Conceptual Models: NDT proposes a strategy to systematically derive both partial conceptual models (an UML class diagram) and partial navigational models from requirement specification. This strategy is applied in the analysis phase. The way in which these conceptual models are derived from requirements is detailed in [18]. The derivation algorithm is implemented by the NDT tool, which allows us to automatically apply it. The main drawback of this approach is that the obtained conceptual models (and specially the navigational one) are defined with the main purpose of analyzing and validating requirements. In this context, only basic conceptual primitives are systematically derived.

To overcome this drawback, the authors of NDT together with the authors of UWE have developed an UML profile for web requirements. Thus, in recent works related to NDT [43], how it fits WebRE has been discussed. In this context, the model-to-model transformations presented in these works16 for obtaining conceptual models from requirement specification can be applied to NDT-based requirement specification. The conceptual model that is obtained however is the one proposed by UWE. The main drawback of this approach is that mode-to-model transformations are still not supported by a tool.

7.10 Proposals of New Development Processes for Web Applications

The approaches presented above are mainly focused on providing mechanisms such as models or tools that support the development of Web applications. Other approaches, however, have focused their efforts on providing new development processes for Web applications, indicating which activities must be performed but recommending the use of existing techniques for supporting these activities. For this type of approaches it is important to mention the following ones:

OO/Pattern Approach: This approach to hypermedia collection design was presented by Thomson, Greer and Cooke in 1998 [18]. It proposes to use both, an OO-design and the application of patterns for the navigational and presentational design. The use of patterns has well-known advantages, such as that the process is well defined, documentation can be reused and maintenance is easy. This approach is divided into six phases: use case design, conceptual design, collaboration design, dictionary definition, navigational design and implementation. Requirements are handled in the first phase where the use of use case diagrams is prescribed to specify the application requirements.

HFPM: Hypermedia Flexible Process Modeling (HFPM) was developed by Luis Olsina in 1998 [18]. This approach proposes thirteen phases for handling aspects such as requirements modeling; project planning, conceptual and navigational modeling, abstract interface modeling, validation, product quality assurance, documentation, etc.

As far as the specification of web application requirements, it is proposed to be performed by means of:

1. A problem description. HFPM indicates that natural language can be used in this description. This description constitutes a textual requirement specification.
2. A use cases model in order to refine the textual requirements describe above.
3. A glossary model. It is used to extract essential problem domain keywords. HFPM proposes to describe these keywords in a classified list written in natural language.

4. A user interface model created by using sketches or prototypes. This model is used in the presentation of drafts to the customer.

RNA: The Relationship Navigation Analysis (RNA) approach was developed by Joonhee Yoo and Michael Bieber in 1998 [7]. This approach is based on the analysis of relationships among the different elements of interest in the Web application domain. This analysis of relationship can help developers to correctly design the navigational structure (nodes connected through links) of Web applications. The development process of this approach is divided into five phases:

1. *Stakeholder Analysis*: The purpose of this phase is to identify the application's audience (stakeholders). Furthermore, analysts also identify and understand the tasks that each type of user might want to perform within the system.
2. *Element Analysis*: In this phase, analysts list all the potential elements of interest in the application. RNA classifies these elements into two types depending on the system under analysis. For a new system, these elements may include all subsystems within the application; all processes that users conduct within the application; all internal processes within the application; and all operations that could be invoked. For existing systems (when a legacy system for the World Wide Web is being reengineered), these elements may include all types of items displayed in any on-line display (information screens, forms, documents, and any other type of display), as well as the screens, forms and documents themselves.
3. *Relationship and Meta-Knowledge Analysis*: In this phase, analysts identify relationships for each element of interest in order to obtain a schema of the Web application. RNA proposes a set of predefined types of relationships that need to be considered in this analysis.
4. *Navigation Analysis*: In this phase, the navigational aspects of a Web application are defined from the relationships identified between the elements of interest.
5. *Relationship and Meta-Knowledge Implementation Analysis*: In this phase, analysts take decisions about how the information identified in the previous phases is implemented in a specific technology

Design-Driven Requirement Elicitation Approach: This approach is part of the Web application design-driven process introduced by David Lowe and John Eklund in 1999 [39]. They propose a generic process of Web development that is both iterative and utilizes design prototypes to assist the client in exploring possible solutions and hence formulating requirements.

This process presents three phases (evaluation, specification and building) that are iteratively performed in two different cycles: exploration and build. In the first cycle, developers repeatedly build prototypes and explore them with the client, obtaining feedback and distilling from this information on the client needs in order to progressively build a specification. Once a sufficient understanding of the client requirements has been obtained, a build contract can be negotiated and then the build cycle commenced. The build cycle is also iterative, and the client understanding and the specification is likely to continue to change. The development in this case is aimed at actually building the system.
It is worth to remark that this approach has been defined from an exhaustive analysis of best practices in the development of Web commercial applications.

GX Web Engineering Method: This method was developed by Jurriaan Souer, Inge Van De Weerd, Johan Versendaal and Sjaak Brinkkemper in 2006 [18]. The GX Web Engineering Method proposes a development process defined by method engineering. This means that phases of existent development methods are selected in order to define a new one that provides support to the Web Application development. In particular, the selected methods are: an old version of GX which is property of a company in the Netherlands, UWE [35] and the Unified Process.
GX is divided in six project phases: Acquisition, Orientation, Definition, Design, Realization and Implementation. For every phase three routes exist: a standard, complex and migration route. Depending on the route different activities are proposed for each phase. The use of each route depends on the complexity of the Web applications. In every route, requirements are described by starting with a goal oriented description. From this description, each route proposes to continue with different types of techniques such as use cases, description of domain terms, description of user types, etc.

7.11 Some Other Approaches

In order to perform the detailed analysis presented in this chapter, other approaches that have not been included because they do not consider RE activities in their development process have been studied. These approaches are mainly focused on providing conceptual models for Web applications. From these approaches it is worth mentioning the following ones:

HDM: The Hypermedia Design Method (HDM) was developed by Garzotto, Paolini and Schwabe in 1993 [37]. It is one of the first methods that were developed to define the structure and interaction in hypermedia applications. HDM is based on the Entity-Relationship model, but extend this model by introducing new primitives to capture what HDM calls the navigational semantics. These primitives are entities (an extended version), components, perspectives, units and links. Currently, HDM is little used because it is based on the structural paradigm and nowadays the object oriented paradigm has gathered strength. However, approaches such as OOHDM, RMM or W2000 were inspired by the ideas of HDM.

RMM: The Relationship Management Methodology (RMM) was developed by Tomas Izsakowitz, Arnold Kamis and Marios Kounfaris in 1995 [35]. It is based on both the E-R model and HDM. Taking these models as sources RMM proposes another model (Relationship Management Data Model, RMDM) in which domain objects, relationships between these objects and navigational mechanisms (such as links, grouping (menus), indexes or guided tours) are described. This approach proposes a development process divided into seven phases: entity relationship design, slice design, navigational design, user interface design, protocol conversion design, run-time behavior, and construction and testing. RMM is supported by the Relationship Management CASE Tool (RMCASE). However, this approach is currently little used because it is based on the E-R paradigm.

MAC-WEB: The Mac-Web Hypermedia Development Environment was developed by Nanard in 1995 [35]. Authors emphasize that communication with users is one of the most important aspects of Web applications. Thus, the development of Web applications is mainly focused on the user interface design. This approach considers this design process from two dimensions: methods steps and mental processes. From the first dimension, formal design techniques to produce the design should be prescribed. Five steps are proposed to do this: concepts elicitation, navigational model, abstract interface, implementation model and testing. From the second dimension, how people actually conduct the design process should be observed. To do this, four steps are proposed: generating material, organizing and structuring, reorganizing and updating and evaluating. Finally, it is worth to remark that designers switch from mental process to method steps as there are not predefined transition rules between the activities or steps.

EORM: The Enhanced Object-Relationship Model (EORM) was developed by Danny Lange in 1996 [18]. This approach is divided into three main phases: analysis, design and implementation. In the first phase, an object oriented model is defined by following the OMT approach. In the second phase, this model is extended by adding additional semantics to the relationships between objects in order to create navigation links. These links are created from a library of links that contains a set of predefined types of links. Finally, in the third phase, the information captured in models is translated into a specific implementation technology. This approach is supported by the ONTOS Studio tool, which have been developed by the author of EORM.

Araneus Project: The work presented in this project was developed by Giasalvatores Mecca, Paolo Atzeni and Paolo Merialdo in 1999 [39]. This work is focused on the management of data bases from a Web environment. This work consists basically in the introduction of hypertext views in the entity-relationship model in order to indicate how data is provided throughout the Web.

Web Composition: This approach was developed by Hans-Werner Gellersen and Martin Gaedke in 1999 [35]. The main goal of this approach is to facilitate the maintenance of Web applications. To do this, they propose the development of Web applications from components. The Web Composition development process starts by defining components that can represent individual links, anchors or other complete resource, such as a HTML page or a Perl Script generating a HTML page. Next, they are composed to develop the Web application.

WUML: Web Unified Modeling Language aims at the methodological support of Web application development with special focus on ubiquity. It was developed by Kappel, Proll, Retschitzegger and Schwinger in 2001 [18]. Authors consider that Web applications should be designed from the start taking into account not only its hypermedia nature, but also the fact that they must run on a variety of platforms such as mobile phones, PDAs, full-fledged desktop computers, and so on. In this context, they propose a model to describe Web applications in which customization plays a main role. Customization is proposed as a uniform mechanism to provide adaptability of a ubiquitous web application.

CHAPTER 8

COMPARATIVE STUDY OF APPROACHES OF REQUIREMENT ENGINEERING FOR WEB APPLICATION

COMPARATIVE STUDY OF APPROACHES OF REQUIREMENT ENGINEERING FOR WEB APPLICATION

In this chapter, the comparative study on two main aspects has been discussed. The first one is the analysis of the types of requirements handled by each methodology. The second aspect is the study of the techniques employed and the phases covered in each approach (requirement elicitation, requirement specification and requirement validation).

8.1 Types of Requirements

Using the classification introduced in Chapter 3, the first objective of this comparison was to establish which types of requirements are treated by each approach. Table 8.1 shows these results for the methodologies briefly described in sections 7.1 to 7.11 and the six types of requirements: data, user interface, navigation, adaptive, transactional and non-functional. Approaches are compared chronologically, which allows us to observe the evolution of the requirement engineering relevance in those methodologies. The first approaches focused mainly on data and user interface requirements. More recently some methodologies have been developed or already existing ones have been extended to manage adaptive, navigation and transactional requirements. The idea of separation of concerns was since the beginning a characteristic of almost all Web methodologies, like HDM [25], OOHDM [51], etc. However, this separation of concerns was only applied to the design and implementation phases of the development process. Nowadays a clear tendency towards a separation of concerns from the very beginning, i.e. already during the requirement elicitation phase can be observed. It is interesting to remark, that the use of different terminology for the same or similar concepts made a comparison study difficult.

Approaches → / Types Of Requirements ↓	OOHDM	WSDM	SOHDM	UWE	WebML	RNA	HFPM	W2000	NDT
Data Requirements	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
User Interface Requirements	Yes	No	Yes	Yes	Yes	Yes	Yes	No	Yes

Navigational Requirements	Yes	No	No	Yes	No	Yes	Yes	Yes	Yes
Adaptive Requirements	No	Yes	No	Yes	Yes	No	No	Yes	Yes
Transactional Requirements	No	No	Yes	No	No	Yes	No	Yes	Yes
Non-Functional Requirements	No	Yes	No	Yes	Yes	No	Yes	No	Yes

Table 8.1 Requirements Handled by Each Approach

8.2 Activities and Techniques

The following table shows the differences regarding the techniques that are used by each methodology in each phase, i.e. during the activities of elicitation; specification and validation in Chapter 3. If a methodology proposes a non-standard technique or a specific technique it will be explicitly indicated. Several conclusions can be obtained from this table. It is possible to indicate that interviewing is the most popular technique during requirements capture. Similarly requirement specification with use cases is the winner for the definition of requirements. The many methodologies handle the capture as part of the definition activity has been shown. In this table discussion of the web methodologies focus on the requirements definition activity has been done. During this activity, the use case technique (the most used one) is applied in different ways. Some methodologies, such as HFPM, apply the original use case technique. However, other approaches, like OOHDM, NDT or UWE, believe that use cases are ambiguous or insufficient for the specification, and so complement this technique with more concrete models, such as UIDs, templates or UML activity diagrams, respectively. This more detailed specification helps to define a more systematic development.

Most of the methodologies consider validation not as relevant as the other two phases: capturing and specification. The validation techniques proposed mainly focus on reviewing the requirements models or the textual descriptions of the requirements.

Comparison According to Requirement Elicitation

Approaches → / Elicitation Techniques ↓	OOHDM	WSDM	SOHDM	UWE	WebML	RNA	HFPM	W2000	NDT
Interviewing	No	Yes	No	Yes	Yes	Yes	No	No	Yes
JAD	No	No	No	No	No	No	No	No	Yes
Brainstorming	No	No	No	No	No	No	No	No	Yes
Concept Mapping	No	Yes	No	No	No	No	No	No	No
Use Case Modeling	Yes	No	No	No	No	No	No	No	No
Questionnaire/ Checklist	No	No	No	Yes	No	No	No	No	No
Sketching and Storyboarding	No	No	No	No	Yes	No	No	No	No

Table 8.2 Techniques Used in Requirement Elicitation

Comparison According to Requirement Specification

Approaches → / Specification Techniques ↓	OOHDM	WSDM	SOHDM	UWE	WebML	RNA	HFPM	W2000	NDT
Natural Language	No	Yes	No	No	Yes	Yes	Yes	No	No
Glossaries	No	No	No	Yes	No	No	Yes	No	Yes
Templates	No	No	No	No	Yes	No	No	No	Yes
Scenarios	No	No	Yes	Yes	No	No	No	No	No
Use Case Analysis	Yes	No	No	Yes	Yes	No	Yes	Yes	Yes
Prototyping	No	No	No	No	No	No	No	No	No

Table 8.3 Techniques Used in Requirement Specification

Comparison According to Requirement Validation

Approaches → / Validation Techniques ↓	OOHDM	WSDM	SOHDM	UWE	WebML	RNA	HFPM	W2000	NDT
Review/Walk-through	No	No	No	Yes	No	No	No	No	Yes
Audit	No	No	No	Yes	No	No	No	No	No
Traceability Matrix	No	No	No	No	No	No	No	No	Yes
Prototyping	No	No	No	Yes	No	No	Yes	No	No

Table 8.4 Techniques Used in Requirement Validation Phases

CONCLUSION AND FUTURE WORK

CONCLUSION AND FUTURE WORK

In this work the analysis of the main approaches published in the literature that provide support for the development of web applications have been discussed. In this analysis, the main focus has been on the requirement specification techniques that these approaches propose. How approaches such us HDM, RMM or ORM belong to web engineering approaches that only provide techniques for the activities of modelling and design of web applications, not for the requirements have been shown. And approaches like OOHDM, WebML, WSDM and OOH are mainly focuses on definition of conceptual models for web application and recommend the use of traditional techniques for requirement specification and focuses on use of use cases diagram have been discussed.

Other approaches such as UWE, W2000 or NDT also use the use cases in the requirement specification. However, these approaches propose some additional techniques for use cases in order to properly describe the navigational requirements. SOHDM is based on scenarios and proposes a notation. This is the only approach that provides specific requirement specification technique for Web applications that is not based on use cases. However, this notation is mainly focused on the specification of transactional requirements and the specification of navigational requirements may result not always easy. Finally, the approaches are compared from different points of view, such as types of requirements handled, and techniques used.

As a result of this study we can claim that there is still a great research potential in the field of requirement engineering for Web applications. It can be observed that the majority of the currently existing Web methodologies focus on design aspects and does not focus its attention on requirement engineering, although the risks of an incomplete or insufficient requirements definition and validation is well-known.

We hope the results presented in this research will help Web developers to select the appropriate requirement 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 requirement engineering, and therefore contribute to improve the quality of the Web applications that are built with these methodologies.

REFERENCES

REFERENCES

[1] Agichtein Eugene, Carlos Castillo, Debora Donato, Aristides Gionis, Gilad Mishne. "Finding high-quality content in social media". WSDM'08 - Proceedings of the 2008 International Conference on Web Search and Data Mining: 183–193, 2008.

[2] Ahlqvist Toni, Back A., Halonen M., Heinonen S. "Social media road maps exploring the futures triggered by social media". VTT Tiedotteita - Valtion Teknillinen Tutkimuskeskus (2454):13, 2008

[3] Al-Rawas A. and Easterbrook S. "Communication problems in requirements engineering: a field study. Proc. Of the first Westminster conference on Professional Awareness in Software Engineering, London. 1996.

[4] Athula Ginige, Sam Murugesan. "The Essence of Web Engineering Managing the Diversity and Complexity of Web Application Development", University of Western Sydney, Australia

[5] Azlena Haron and Shamsul Sahibuddin. The Strength and Weakness of Requirement Engineering (RE) Process, 2010.

[6] Balachander Krishnamurthy, Graham Cormode. "Key differences between Web 1.0 and Web 2.0". First Monday, Volume 13 Number 6, June 2008.

[7] Baresi L., Garzotto F. and Paolini, P. "Extending UML for Modeling Web Application." Probceeding of the 34th Hawaii International Conference on System Science, 2001.

[8] Berners-Lee, Tim, James Hendler and Ora Lassila. "The Semantic Web". Scientific American Magazine. May, 2001.

[9] Cern.info.ch - Tim Berners-Lee's proposal. Info.cern.ch/proposal.html

[10] Christel, Michael G.; & Kang, Kyo C. Issues in Requirements Elicitation (CMU/SEI-92-TR-12). Pittsburgh, PA: software Engineering Institute, Camegie Mellon University, September, 1992.

[11] Cutter Consortium. "Poor Project Management Number one Problem of Outsourced E-projects", Cutter Research Briefs, November 2000.

[12] Davis A.M."Software Requirements: Objects, Functions and States." Prentice Hall, 1993. Revised Edition, 1993

[13] De Troyer O., Leune C. "WSDM: A User-Centered Design Method for Web Sites", Computer Networks and ISDN systems, Proceedings of the 7th International World Wide Web Conference, pp. 85 - 94, Publ. Elsevier, Brisbane, Australia, 1998.

[14] Dean Leffingwell, DON AUTOR WIDRIG. Managing software requirements: a unified approach. Addison-Wesley Professional, 2000.

[15] DeMarco T. Structured Analysis and System Specifications, Yourdon Press, Englewood Cliffs, N.J, 1979.

[16] Deshpande Y. and Hansen S. "Web Engineering: Creating a Discipline among Discipline, IEEE Multimedia, Special issues on Web Engineering, Vol 8, 2001.

[17] E. Kazmierczak. "Requirements Engineering", The University of Melbourne, 2003

[18] Escalona M.J. and Koch N. "Requirements engineering for web applications: A comparative study." Journal on Web Engineering, Rinton Press, 2004.

[19] G. Rossi, D.Schwabe, C.J.P Lucena, D.D. Cowan "An Object-Oriented Model for Designing the Human-Computer Interface of Hypermedia Applications". 1995

[20] Garrigos I., Gomez J., Barna P. and Houben G.J."A Reusable Personalization Model in Web Application Design. In 2nd Workshop on Web Information Systems Modeling (WISM 2005), in conjunction with ICWE, Australia, 2005.

[21] Garzotto F., Schwabe D. and Paolini P. "HDM - A Model Based Approach to Hypermedia Application Design". ACM Transactions on Information Systems, Vol. 11, No. 1, January, 1993

[22] Global Harmonization Task Force - Quality Management Systems – Process Validation Guidance (GHTF/SG3/N99-10:2004) (Edition 2) page 3

[23] Gomez J., Cachero C. and Pastor O. "Extending a Conceptual Modeling Approach to Web Application Design." In Proc. Of the 12th international Conference on Advanced information systems engineering. LNCS1789, 2000.

[24] Gotel O. and Finkelstein A. "An Analysis of the Requirements Traceability Problem.", Proceedings of the First IEEE International Conference on Requirements Engineering, Colorado springs, 18-22 April, 1994.

[25] Grady Autor Booch, James Rumbaugh, Ivar Jacobson. "The Unified Modeling Language User Guide.", Addison-Wesley, 1999

[26] Heeseok Lee, Choongseok Lee, Cheonsoo Yoo. "A Scenario-Based Object-Oriented Methodology for Developing Hypermedia Information Systems.", HICSS (2) 1998

[27] IEEE Recommended Practice for Software Requirements Specifications -830-1998

[28] IEEE Standard for System and Software Verification and Validation-1012

[29] Ivar Jacobson, Grady Booch, James Rumbaugh. “The Unified Software Development Process”, February 1999.

[30] Japanese Pharmacopoeia, the 15th edition. 29 June 2012.

[31] Jim Conallen. “Building Web Applications with UML”, Dec 7 1999.

[32] Karlsson Lena. “Requirements engineering challenges in market-driven software development – An interview study with practitioners”, 2007.

[33] Kotonya, G. and Sommerville, I. “Requirements Engineering: Processes and Techniques”, Chichester, UK: John Wiley and Sons, 1998.

[34] Nigel Morgan, Graham Jones, Ant Hodges. "Social Media". The Complete Guide to Social Media from the Social Media Guys, 12 December 2012.

[35] Nora Koch. “Transformation Techniques in the Model-Driven Development Process of UWE.” 2nd Model-Driven Web Engineering 2006 Workshop (MDWE'06), Palo Alto, USA. ACM DL, July 2006.

[36] O'Reilly Tim. "What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software". O'Reilly Media. September 30, 2005.

[37] Paul Graham. "Web 2.0". November, 2005.

[38] Philip Sallis, Graham Tate, and Stephen McDonell. “Software Engineering.” Addison Wesley Publishing Co., 1995.

[39] Piero Fraternalli, Aldo Bongio and Stephano Ceri. “Web Modeling Language (WebML): a modeling language for designing web sites”, June 2000.

[40] Ramos Rowel, Kurts Alfeche. “Requirements Engineering: A good practice guide”, John Wiley and Sons, 1997

[41] Saiediana H., Daleb R. “Requirements engineering: making the connection between the software, developer and customer”, 2000.

[42] Schwabe D., Rossi G. and Barbosa S. "Systematic Hypermedia Application Design with OOHDM". Proceedings of Hypertext '96, Washington, March (New York: ACM Press), 1996.

[43] Segura S., Benavides D., Ruiz-Cortes A. and Escalona M.J. “From Requirements to Web System Design: An Automated Approach using Graph Transformations.” DSDM, 2007.

[44] Shams Ul Arif, Qadeem Khan, S.A.K. Gahyyur, “Requirements Engineering Processes, tools/technologies, and methodologies “

[45] Stevens R., Brook P., Jackson K. & Arnold S. “Systems Engineering: Coping with Complexity” Prentice Hall Europe, 1998.

[46] W3C Semantic Web Activity". World Wide Web Consortium (W3C). November 7, 2011.

[47] XML and Semantic Web W3C Standards Timeline, 2012.

[48] Young r. r., “Effective requirements practices”, boston, ma: Addison-wesley, 2001.

[49] http://blog.nwcadence.com/the-problem-with-requirements-engineering-tools/. Retrieved on 23 April 2013.

[50] http://blogs.gartner.com/anthony_bradley/2010/01/07/a-new-definition-of-social-media/. Retrieved on 3 April 2013.

[51] http://en.wikipedia.org/wiki/Verification_and_validation. Retrieved on 5 May 2013.

[52] http://requirements.seilevel.com/blog/2011/12/is-social-media-useful-for-requirements-elicitation.html. Retrieved on 1 May 2013.

[53]http://scholar.googleusercontent.com/scholar?q=cache:v9BjMTEh0rQJ:scholar.google.com/+web+2.0+definition&hl=en&as_sdt=0,33&as_vis=1. Retrieved on 3 April 2013.

[54] http://www.freelock.com/newsletter/10-problems-web-development-projects-and-how-weve-solved-them. Retrieved on 23 April 2013.

[55] http://www.sas.com/software/customer-intelligence/social-media-analytics/#section=1. Retrieved on 5 May 2013.

[56]http://www.techrepublic.com/article/five-common-errors-in-requirements-analysis-and-how-to-avoid-them/6146544. Retrieved on 3 April 2013.

[57]http://www.yourmaindomain.com/web-articles/use-of-internet.asp. Retrieved on 5 May 2013.

APPENDIX A

REQUIREMENT ENGINEERING OF A WEB APPLICATION USING NDT METHOD

Requirement Engineering of a Web Application using NDT Method

A.1 Introduction

Appendix comprised of a detailed case study of a web application has been done where the contributions of this thesis put into practice. In this case study, requirement engineering of an E-Commerce web application like any other e-commerce site is done. We have chosen an E-commerce application because this type of Web applications are focused on both providing great amount of information to users and providing them with complex functionality in order to allow the purchase of products. These aspects allow us to properly show the main characteristics of our approach. We have used an E-commerce application similar to any other e-commerce application in order to facilitate its understanding since these web applications are one of the most used web applications.

Main characteristics of the case study are as follows:

- The case study is an E-commerce application that must support the online purchase of books, and CDs.
- This e-commerce application must allow users which are initially connected to the system as visitors to access information about the different products that exists in the catalogue.
- For each book, users must be able to know the title, the publishing year, edition, the front cover, the price, some comments and the list of chapters.
- For each CD, users must be able to know the title of CD, the name of the artist, the front cover, the price, the version, and the year in which it was released, a summary, and the cover.
- The e-commerce application must provide users with a shopping cart where they can add the products to be purchased; users can add as many products as they want.
- The shopping cart must be always available to users in order to allow them to manage it, users must be able to access the shopping cart to inquiry the products that have been added.
- Users must also be able to remove products or modify the amount of product units that have been added.
- When products are added to the shopping cart the system must automatically update the product stock, i.e. the quantity of product units that there are in the warehouse must be updated after adding the product to the shopping cart.

- System must also register the times that a product is purchased as well as the client profiles that usually purchase this product.
- Once users have selected the products to be purchased by adding them to the shopping cart the E-commerce application must allow them to create and send a purchase order. To do this, users must identify themselves as registered customers.
- To register, user must provide the following information: name and surname, address, city where they live as well as the country and the postal code, telephone number, and a unique login and a password.
- When the system asks users for identification they must introduce the login and the password.
- Administrators must be allowed to manage all the information about products.
- Administrators must be able to introduce new products in the catalogue, modify the information of the existent products or deleting products.
- Administrators must also be able to manage the information of the registered customers. They must be able to modify customer's information and delete them.

The rest of this appendix is organized as follows: Section A.2 describes the task-based requirements model that specifies the requirements of the E-commerce Application.

A.2 Requirements Model

Next, we show the task-based requirements model of the case study. First, we describe the statement of purpose. Next, we explain how task users are identified and described. Finally, we specify the data that the system must store.

1. Statement of Purpose

The statement of purpose of the case of study is the following:

The web application under development is an E-commerce Application. The main goal of the system is to provide support for the on-line purchase of products. To achieve this, the user must be able to consult information about the products, add them to a shopping cart and send purchase orders. Furthermore, tools for the management of information must be provided.

2. Description of User Tasks

Next, we introduce the task taxonomy of the case of study as well as the description of elementary tasks.

2.1 Tasks Classification

The Tasks Classification of the e-commerce example is shown in Figure A.1. It is described as follows:

- From the statement of purpose we extract the most general task that is *Purchase Products*.

- The task *Purchase Products* is decomposed by means of a structural refinement (solid line) into two tasks:
  (1) *Buy Products* which involves the needs related to the process of buy products and
  (2) *Manage Information* which involves the needs related to management activities.

- The task *Buy Products* is decomposed by means of a temporal refinement (dashed line) into *Collect Products* and *Checkout*. The relation between them is enabling with information exchange. Thus, the products should be collected into the shopping cart before checkout. The information that needs to be exchanged is the collected products.

- In order to *Collect Products*, users add them to the shopping cart. During this process, users can inspect the shopping cart when they consider opportune.

- Thus, *Collect Products* is decomposed into Add Product to Shopping Cart and Inspect Shopping Cart. The relation between the two tasks is suspend-resume, which indicates that *Add Product* to Shopping Cart may be interrupted at any point by Inspect Shopping Cart (this is not mandatory, see how the second task is defined as an optional task). After each interruption, the task *Add Product* to Shopping Cart is resumed.

- Add Product to Shopping Cart is an iterative task. Then, it can be performed so many times the user needs. It is decomposed by means of a structural refinement into the tasks *Add CD, Add Software*, and *Add Book*. Then, when the user is adding a product to the shopping cart he/she is adding a CD, a Software product or a Book.

- The task *Manage Information* is decomposed by means of a structural refinement into *Manage Products* and *Manage Customers*. Users can manage information about either products or customers.

- Finally, the task *Manage Products* is decomposed by means of a structural refinement into *Manage CDs, Manage Software* and *Manage Books*. Users can manage information about CDs, software or books.

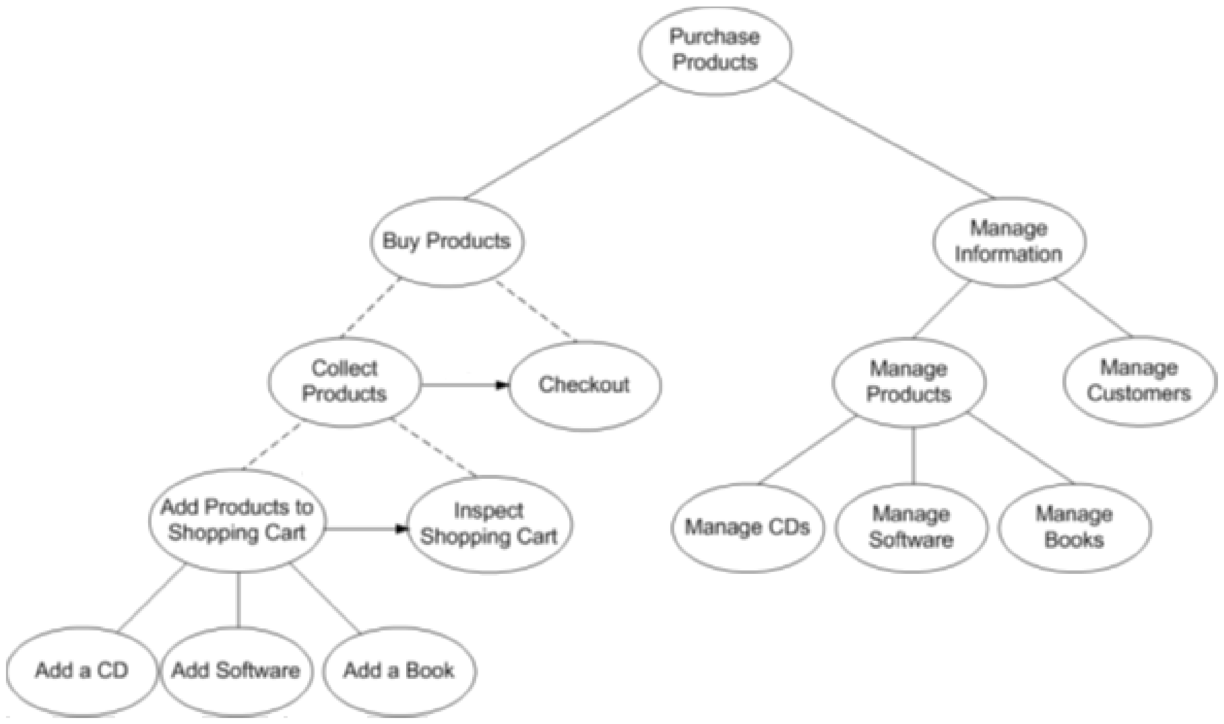


Figure A.1 User Tasks

2.2 Description of Elementary Tasks

Next, the description of some elementary tasks of the user task also called as task taxonomy. In particular, the introduction of the following elementary tasks: *Add CD, Add Book, Inspect Shopping Cart, Checkout* and *Manage Software*. For each of these elementary tasks, we present first its characterization and next the description of the elementary task performance.

Add Book

This task describes how users must add books to the shopping cart. Figure A.2 shows its characterization. According to this figure, the goal of the task is for the user to add a book to the shopping cart; the task can be completed by visitors and customers. As an estimate, the task is going to be completed 100 times per hour. Finally, no additional constraints are defined.

Task	Add Book
Goal	The user add book to shopping cart
Users	Visitors and customers
Frequency	100 times per hour
Additional constraints	-------

Figure A.2 Characterization of Task Add Book

Figure A.3 shows how the elementary task *Add Book* must be performed. This task starts with an Output where the system provides the user with a list of books categories. From this list, the user can select a category. If the category has subcategories, the system provides again the user with a list of sub categories. If the selected category does not have subcategories the system informs about the books of the selected category by means of an Output. The user can perform two actions from this last output:

(a) Select a book, and then the system provides the user with a description of the selected book.

(b) Activate a search operation, and then the system performs a system action which searches for the books of an author. To do this, the user must introduce the author by means of an Input. If the search returns only one book, the system provides the user with its detailed description. Otherwise, the system provides the user with a set of books.

Finally, when the user has obtained a book description then he can:

(a) Select the author of the book, and then the system provides the user with detailed information about the author. In this case, it is mandatory that the user returns to the book description in order to achieve the task goal to add a book to the shopping cart.

(b) Activate the Add to Cart operation which is an operation that allows users to manipulate entities. When users activate this operation the system performs an action that adds the selected book to the shopping cart. Next, the system updates the stock23 and finally the task finishes.

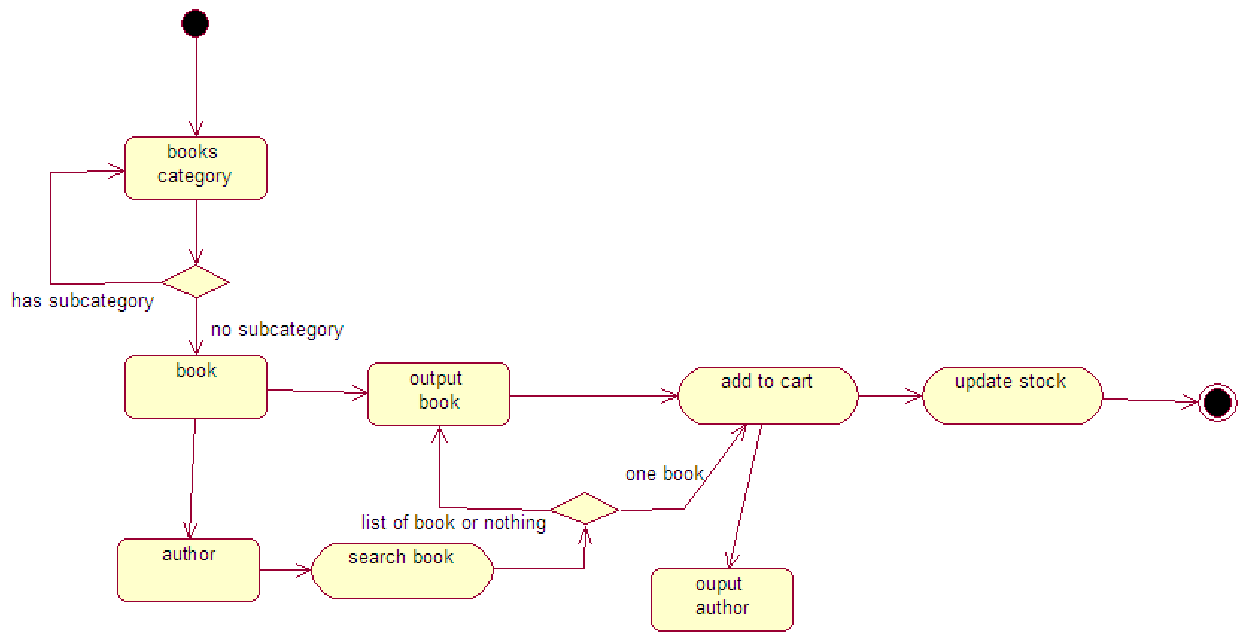


Figure A.3 Description of Performance of Task Add Book

Add CD

This task describes how users must add CDs to the shopping cart. Figure A.4 shows its characterization. According to this figure, the goal of the task is for the user to add a CD to the shopping cart; the task can be completed by visitors and customers. As an estimate, the task is going to be completed 100 times per hour. Finally, no additional constraints are defined.

Task	Add CD

Goal	The user add CD to shopping cart
Users	Visitors and customers
Frequency	100 times per hour
Additional constraints	-------

Figure A.4 Characterization of Task Add CD

Figure A.5 shows how the elementary task Add CD must be performed. The user can start this task by activating two different search system actions:

(A) One that searches for CDs using a cd property as a search criterion or

(B) Another that searches for CDs using a subject as a search criterion. As a result of the first type of search from a cd property, a list of CDs is provided to users.

As far as the second type of search from a cd subject there are two possibilities:

(a) If the subject introduced by users exists and CDs are found, the list of CDs that belongs to the subject is provided to users; otherwise,

(b) Users are provided with a list of subjects. From this list, users can select one and then they access the list of CDs that belong the selected subject. From the list of CDs accessed by any search users can select one cd and then the system provides them with a detailed description of the selected cd. From this description, users can activate the “Add to Cart” operation which is an operation that allows users to manipulate entities. When users activate this operation, the system performs an action that adds the selected cd to the shopping cart. Next, the system updates the stock and finally the task finishes.

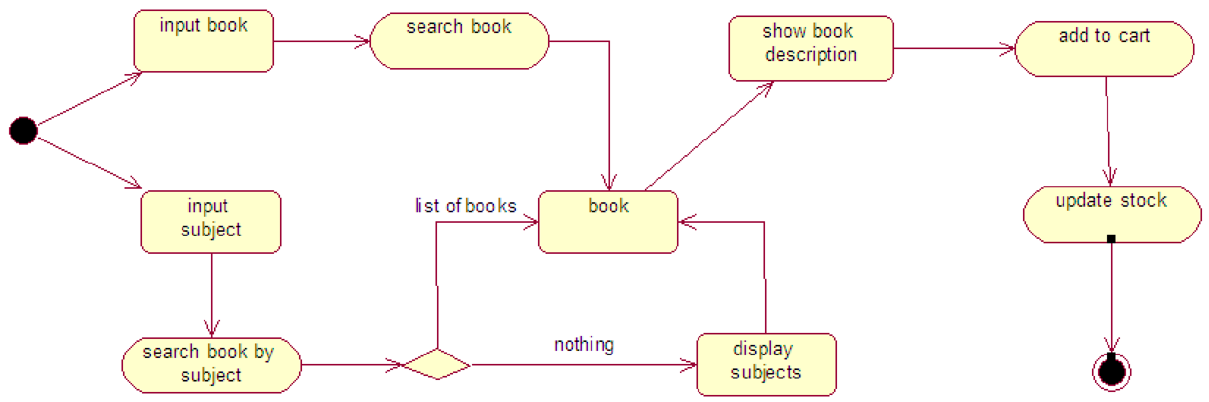


Figure A.5 Description of Performance of Task Add Book

Inspect Shopping Cart

This task describes how users can inspect the shopping cart. Figure A.6 shows its characterization. According to this figure, the goal of the task is for the user to manage the items that have been added to the shopping cart; the task can be completed by visitors and customers. As an estimate, the task is going to be completed 40 times per hour. Finally, no additional constraints are defined.

Task	Inspect shopping cart
Goal	The user manage the items that have been added to shopping cart
Users	Visitors and customers
Frequency	50 times per hour
Additional constraints	-------

Figure A.6 Characterization of Task Inspect Shopping Cart

Figure A.7 shows how the elementary task Inspect Shopping Cart must be performed. This task starts with an output where the system provides users with a list of items. For each item, users can activate two system actions: delete item and modify item. If the first one is activated, the system removes the corresponding item from the shopping cart. If the second one is activated, the system modifies the information associated to the corresponding item. To do this, users must introduce the new information by means of an input. Furthermore, users can select one item from the list and then the system provides users with a description of the product associated to the item. In this case, it is mandatory that the user returns to the list of items in order to achieve the task goal (to manage items of the shopping cart).

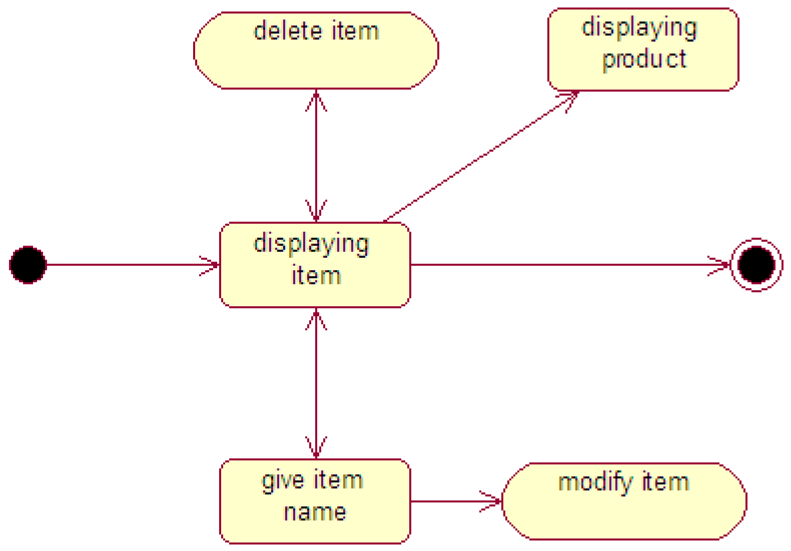


Figure A.7 Description of Performance of Task Inspect Shopping Cart

Checkout

This task describes how users can checkout once they have collected products. Figure A.8 shows its characterization. According to this figure, the goal of the task is for the user to send a purchase order; the task can be completed only by customers. As an estimate, the task is going to be completed 25 times per hour. Finally, an additional constraint is defined. This constraint indicates that a security protocol must be implemented in order to perform this task.

Task	Checkout
Goal	User sends the purchase order
Users	Customers
Frequency	25 times per hour
Additional constraints	Security protocol must be implemented to perform this task

Figure A.8 Characterization of Task Inspect Shopping Cart

Figure A.9 shows how the elementary task Checkout must be performed. This task starts with an Output where the system provides the user with information about its purchase order. In order to follow with the task the user needs to be a registered customer. Thus, from this Output IP, the user can: (1) identifies itself as a register customer in order to follow with the task or (2) register itself as a customer. If the identification is successful (the user is a registered customer) the task follows by sending the purchase order. Otherwise (the user is not a registered customer) the user accesses again the Output IP that provides the user with the order information. From this Output IP, the user can register itself as a customer or try to login again.

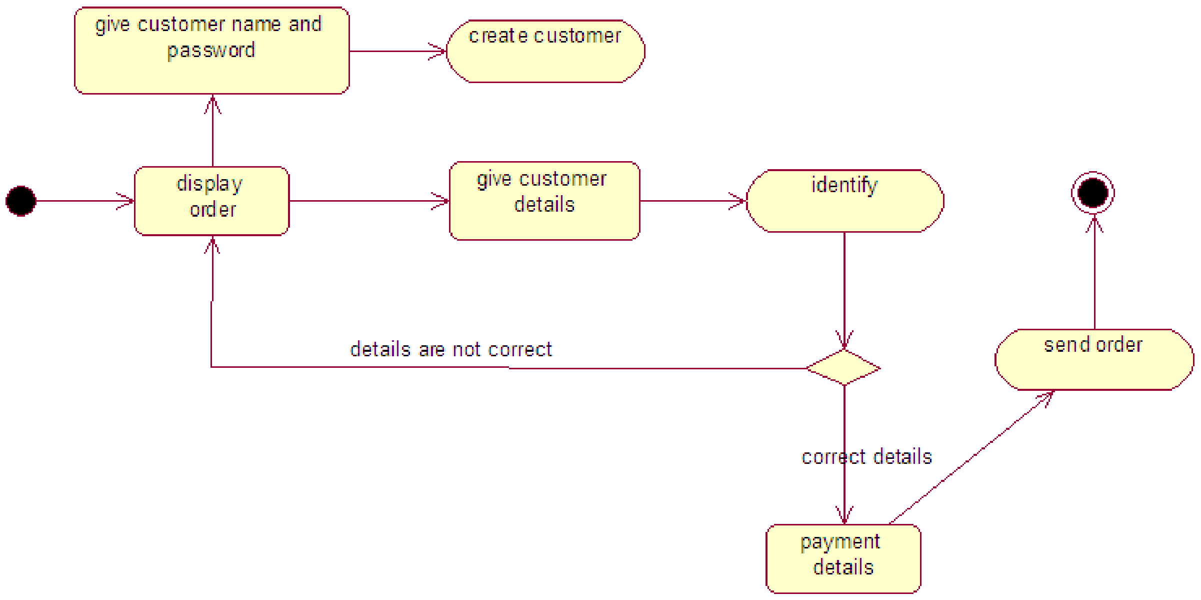


Figure A.9 Description of Performance of Task Checkout

2.3 Description of the System Data

Next, we show how the information requirements are specified. We first introduce the information templates. Next, the exchanged data templates are presented.

2.3.1 Information Templates

We define an information template for each entity identified in the task performance descriptions. As representative example we show the following: CD, Artist, Music Category, Product, Item, and Order.

CD

The information template that is associated to the entity CD is shown in Figure A.10. The information that the system must store about a CD is (see the specific data section): the CD title, the recording year, the artist that has recorded the CD, the list of songs, some comments about the CD, the front cover, the price, the number of times that the CD has been bought, the profiles of the customers which usually purchase it, the number of CD units that there are in stock and the music categories to which belong the

CD. All these features present a simple nature except from the artist that has recorded the CD and the music categories.

Identifier	Id1		
Entity	CD		
Specific data	Name	Description	Nature
	Title	Title of the CD	String
	Year	Recording year of the CD	Number
	Artist	Artist that has recorded	Id2
	Songs	Songs of the CD	String list
	Comments	A brief commentary of the cd	Text
	Front cover	Front cover of cd	Image
	price	Price	Currency
	Purchase times	Times that cd has been purchased	Number
	Client profile	Profile of clients purchased the cd	String list
	Stock	Quantity of units in stock	Number
	Music category	Categories of which belong the cd	Id3

Figure A.10 Information Template Associated to Entity CD

Artist

The information template that is associated to the entity Artist is shown in Figure A.11. The information that the system must store about an Artist is the following: the name of the artist, the nickname, the date of the artist's birth, the country where the artists was born, the record label with which the artist has signed a deal and the artist's personal web page. All these features present a simple nature.

Identifier	Id2		
Entity	Artist		
Specific data	Name	Description	Nature
	Name	Name of artist	string

	Nickname	Nickname of artist	String
	Birth date	Date of birth of artist	Date
	Country	Country in which artist born	String
	Label	Record label with which the artist has signed a deal	String
	Webpage	Webpage of artist	url

Figure A.11 Information Template Associated to Entity Artist

Music Category

The information template that is associated to the entity Music Category is shown in Figure A.12. The information that the system must store about a Music Category is the following: the name of the music category, a description about it, a description of cultural aspects from which the music category is originated, a list of the instruments typically used in this kind of music and a list of subcategories. All these features present a simple nature except the last one that is defined as a list of subcategories. Each of these subcategories is described by the same information template shown in Figure A.12.

Identifier	Id3		
Entity	Music category		
Specific data	Name	Description	Nature
	Name	Name of music category	String
	Description	Description of music category	Text
	Cultural origins	Description of origins of music category	String
	Typical instruments	List of instruments used for playing this music	String list
	Subcategory music	Music subcategories derived	Id3

Figure A.12 Information Template Associated to Entity Music Category

Item

The information template that is associated to the entity Item is shown in Figure A.13. The information that the system must store about an Item is the following: the product selected by the user, the quantity of product units that have been selected and the total price of the selected units. The two last features have a simple nature. The first one has a complex nature that is described by the information template in Figure A.14.

Identifier	Id4		
Entity	Item		
Specific data	Name	Description	Nature
	Product	Product included in item	Id5
	Quantity	Quantity of products included in item	Number
	Total price	Total price of item	Currency

Figure A.13 Information Template Associated to Entity Item

Product

The information template that is associated to the entity Product is shown in Figure A.14. When we try to describe the specific data associated to this entity we realize that it depends on whether the product is a CD, a Book or Software. Thus, the entity Product is a super-type of the entities CD, Book and Software. In this context, depending on the product type, the specific data section associated to the entity Product is the specific data section that is associated to the entities CD, Book or Software.

Identifier	Id5	
Entity	Product	
Specific data	Super set of:	
	CD	Id1
	Book	Id6

Figure A.14 Information Template Associated to Entity Product

Order

The information template that is associated to the entity Order is shown in Figure A.15. The information that the system must store about an Order is the following: the items that are included in the order, the customer who purchase the items, the total price of the purchased items, and the credit card used to pay. Only the total prices have a simple nature. The rest of features present a complex nature (described in the information templates Id8, Id9 and Id12).

Identifier	Id7		
Entity	Order		
Specific data	Name	Description	Nature
	Items	List of items are purchased	Id5
	Customer	Customer that purchased item	Id8
	Total price	Total price of purchase	Currency
	Credit card	Credit card used to pay	Id9

Figure A.15 Information Template Associated to Entity Order