

Requirements Change Management based on Web Usage Mining

Jorge Manuel Esparteiro Garcia

January 2016

Scientific Supervision by

PhD, Ana Cristina Ramada Paiva, Assistant Professor
Departamento de Engenharia Informática

In partial fulfillment of requirements for the degree of
Doctor of Philosophy in Informatics Engineering
by the ProDEI Doctoral Programme

Contact Information:

Jorge Manuel Esparteiro Garcia
Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia Informática

Rua Dr. Roberto Frias, s/n
4200-465 Porto
Portugal

Tel.: +351 22 508 1400

Fax.: +351 22 508 1440

Email: jorge.garcia@fe.up.pt

URL: <http://portal.ipvc.pt/images/ipvc/esce/docentes/jgarcia/>

Jorge Manuel Esparteiro Garcia

“Requirements Change Management based on Web Usage Mining”

Copyright © 2015 Jorge Esparteiro Garcia. All rights reserved.

...to my sweet daughter Carolina

...to my lovely wife Ana

This page was intentionally left blank.

"The consequences of things are not always proportionate to the apparent magnitude of those events that have produced them."

Charles Caleb Colton

This page was intentionally left blank.

Abstract

In recent years, the use of the World Wide Web (WWW) has had a huge growth and there is a greater variety of web applications with an increasing importance in society and in supporting the development to all kinds of business.

Often, most of websites are providing support services that must be maintained and improved over time. This maintenance and upgrade can be difficult because frequently the requirements are no longer actual and/or often not even exist documented. Furthermore, it can also be difficult to assess what are the most critical features in order to define the changes to implement first (in the case of several requests).

Websites are increasingly monitoring usage data, and this type of information is increasingly abundant as the meta-data that is possible to extract from the web navigation. However, commonly, the data about the usage of web sites is used only for reporting and for analysing the usage traffic not adding any value to the strengthening and improvement of web sites.

Extracting and analysing the information about the usage of the websites can help identify improvements on the website and help to maintain requirements updated which can be a contribution to the overall quality of the service provided.

Along the research work described in this dissertation, we develop an approach and supporting tool (called REQAnalytics) to collect the information about the usage of the website, to analyse it and to build reports with recommendations that can, hopefully, contribute to increase the quality of the requirements specification and the website itself.

The contributions of this research work are: a set of rules to analyse usage data; an high-level mapping tool that through a web-based application, maps

the functional requirements of the software requirements specification with the web pages and elements of the website; a novel approach to Requirements Management through a Recommender System, that collects information on the usage of a website, relates that information back to the requirements and generates reports with recommendations to the requirements specification (e.g. change the requirements priority, create new requirements, remove existing requirements).

In order to evaluate the overall approach, we performed evaluation of case studies over existing websites. The results showed that REQAnalytics can produce reports in a language closer to the business, recommend new workflows, identify functionalities to create and remove, and, ultimately, give support to the maintenance of requirements of the website being analysed.

Resumo

Nos últimos anos, a World Wide Web (WWW) tem tido um enorme crescimento e existe uma maior variedade de aplicações *web* com uma importância crescente na sociedade e no apoio ao desenvolvimento de todos os tipos de negócio.

Muitas vezes, a maioria dos sítios *web* oferecem serviços de apoio que devem ser mantidos e melhorados ao longo do tempo. Esta manutenção e atualização pode ser difícil porque muitas vezes os requisitos não estão atualizados e / ou muitas vezes nem sequer podem já existir documentados. Além disso, também pode ser difícil avaliar quais são as características críticas para determinar as alterações a serem implementadas primeiro (no caso de várias solicitações).

Os sítios *web* estão monitorizando cada vez mais os seus dados de utilização, e este tipo de informação é cada vez mais abundante, como por exemplo os meta-dados que são possíveis extrair a partir da navegação *web*. No entanto, normalmente, os dados de utilização dos sítios *web* são utilizados apenas para fins de comunicação e marketing e para a análise do tráfego de utilização, não acrescentando qualquer valor para a melhoria do sítio *web*.

A extração e a análise da informação da utilização dos sítios *web* conseguem ajudar a identificar melhorias e ajudam a manter o sítio *web* e os seus requisitos, o que pode ser uma contribuição para a qualidade global do serviço prestado.

Ao longo deste trabalho de investigação, desenvolvemos uma abordagem e uma ferramenta de apoio (chamada REQAnalytics) para recolher a informação de utilização do sítio *web*, analisá-la e gerar relatórios com recomendações que irão, esperamos, contribuir para aumentar a qualidade da especificação de requisitos e do próprio sítio *web*.

As ferramentas desenvolvidas durante este trabalho de investigação foram

aplicadas sobre diferentes sítios *web*, com o objetivo de avaliar a hipótese de investigação. Os resultados da abordagem proposta juntamente com a análise de cada sítio *web* podem fornecer relatórios mais legíveis numa linguagem mais perto de negócio, indicar novos fluxos de trabalho, identificar funcionalidades que podem ser removidas e suportar a manutenção do sítio *web*.

As contribuições deste trabalho de investigação serão: um conjunto de regras para analisar os dados de utilização; uma ferramenta de mapeamento de alto nível que através de uma aplicação baseada na *web*, mapeia os requisitos funcionais com as páginas e os elementos do sítio *web*; uma nova abordagem para gestão de requisitos através de um sistema de recomendação, que recolhe a informação de utilização de um sítio *web*, relaciona essa informação com os requisitos e gera relatórios com recomendações à especificação de requisitos (e.g. alterar a prioridade dos requisitos, criar novos requisitos, remover requisitos existentes).

Com o objetivo de avaliar a abordagem global, realizámos a avaliação de casos de estudo em sítios de *web* existentes. Os resultados mostraram que o REQAnalytics consegue produzir relatórios numa linguagem mais próxima do negócio, recomendar novos *workflows*, identificar funcionalidades a criar e a remover, e, finalmente, dar suporte à manutenção dos requisitos do sítio *web* em análise.

Contents

Abstract	i
Resumo	iii
Acknowledgments	xvii
1 Introduction	1
1.1 Problem and Motivation	3
1.2 Research Questions and Goals	5
1.2.1 Research Questions	6
1.2.2 Research Goals	6
1.3 Research Strategy	7
1.4 Contributions	9
1.5 Results	9
1.6 Structure of Thesis	10
1.7 Publications	12
I Background and State of the Art	15
2 Requirements Engineering	17
2.1 Introduction	18
2.1.1 Requirement Definition	18
2.1.2 Taxonomies	19
2.1.3 Requirements Types	20
2.1.4 Other Requirements Taxonomies	22

2.2	Requirements Engineering	22
2.2.1	Requirements Elicitation	24
2.2.2	Requirements Specification	24
2.2.3	Requirements Validation	24
2.2.4	Requirements Management	24
2.2.5	Requirements Evolution	27
2.2.6	Traceability	27
2.3	Requirements Management Tools	30
2.3.1	IBM Rational Doors	31
2.3.2	Rational RequisitePro	31
2.3.3	CaliberRM by Borland	31
2.3.4	Requirements Management Approaches	32
2.4	Discussion	37
2.5	Summary	37
3	Web Usage Mining	39
3.1	Web Mining	40
3.2	Web Usage Mining	41
3.2.1	Web Server Log File	41
3.2.2	Web Usage Mining Tasks	42
3.2.3	Preprocessing	42
3.2.4	Pattern Discovery	43
3.2.5	Pattern Analysis	44
3.3	Web Usage Mining Approaches	44
3.4	Web Analytics	45
3.4.1	Tools	46
3.5	Discussion	47
3.6	Summary	49
4	Recommender Systems	51
4.1	Recommender Systems	52
4.2	Collaborative filtering	52
4.3	Content-based filtering	53
4.4	Hybrid Recommender Systems	53

4.4.1	Taxonomy of Hybrid Recommendation Systems	53
4.5	Recommender Systems for Software Engineering	55
4.6	Recommender Systems Approaches	56
4.7	Discussion	58
4.8	Summary	59
 II Research Approach and Solution		61
5	Research Problem	63
5.1	Research Challenges	64
5.2	Thesis Statement	65
5.3	Research Questions	65
5.4	Research Goals	66
5.4.1	Primary Goal	66
5.4.2	Secondary Goals	67
5.5	Validation Methodology	68
5.6	Summary	68
6	REQAnalytics	71
6.1	Overview	73
6.2	Architectural Design	75
6.3	Web Analytics Tool Selection	76
6.3.1	Open Web Analytics	78
6.4	Mapping Tool	79
6.4.1	What Information?	80
6.4.2	How capture the information?	81
6.4.3	Development of the tool	81
6.5	Collect Web Usage Data	83
6.6	Analysis of the Data Collected	83
6.7	Generation of Recommendation Report	84
6.8	Components of REQAnalytics	84
6.8.1	Requirements Import	85
6.8.2	Functional Requirements List	85
6.8.3	Details of the Requirement	87

6.8.4	Dashboard Visualization	89
6.8.5	Requirements Dependencies Graph	91
6.8.6	Most used Requirements Navigation Paths	93
6.8.7	Traceability Matrix	93
6.8.8	Requirements Analytics - Statistics and Main Metrics	94
6.8.9	Recommendations to the Software Requirements Specification (SRS)	96
6.9	Summary	101
III Validation and Conclusions		103
7	Validation	105
7.1	Case Study 1: Website of Regional Newspaper	107
7.1.1	Goals	107
7.1.2	Setup	108
7.1.3	Procedure	109
7.1.4	Results	109
7.2	Case Study 2: SIGARRA: Faculty of Engineering of University of Porto Website	113
7.2.1	Goals	113
7.2.2	Setup	114
7.2.3	Procedure	114
7.2.4	Results	115
7.3	Discussion	120
7.3.1	Case Study 1	121
7.3.2	Case Study 2	122
7.4	Summary	123
8	Conclusions	125
8.1	Summary	126
8.2	Main Contributions	127
8.3	Opportunities for Future Research Work	128
8.4	Conclusion	129

Glossary	131
Bibliography	133

List of Figures

2.1	Requirements engineering processes	23
2.2	Major RM activities [Wie03]	25
2.3	Change management workflow process	26
2.4	Bidirectional - Forward and Backward Traceability [Wes06]	29
2.5	Requirement Engineering tools' scores and prices [CNA ⁺ 11]	35
2.6	Screenshot of IBM Rational Doors	36
2.7	Screenshot of CaliberRM	36
3.1	Web Mining Categories	41
3.2	Best Key Practices: Web Analytics Process Guide [BJo8, Chr]	45
3.3	Metrics Categories	46
4.1	Combined mapping of the RE activity, recommendation technique, and type of items recommended [MRE12]	56
6.1	An overview of the Architectural Design of REQAnalytics recom- mender system	75
6.2	Open Web Analytics Dashboard	78
6.3	Functional requirements mapped with the web pages and elements	80
6.4	XSD Schema Diagram of the XML file necessary to import the functional requirements	81
6.5	Mapping tool inside a bookmarklet	82
6.6	Mapping tool inside a bookmarklet to create the traceability links between requirements and the implementation	83

6.7	Screenshot of the component Requirements Import XML of REQ-Analytics.	85
6.8	List of the Functional Requirements previously imported to REQ-Analytics	86
6.9	Screenshot of REQAnalytics with a Tooltip available in the Requirements List view, with a mapping and their HTML element id and HTML element type.	87
6.10	REQAnalytics component Details of Requirement.	89
6.11	Screenshot of REQAnalytics main Dashboard	90
6.12	Requirements Dependencies Directed Graph	91
6.13	Screenshot of Most used Requirements Navigation Paths - Table view	93
6.14	Screenshot of the Traceability Matrix	94
6.15	Screenshot of Requirements Analytics available in REQAnalytics - Statistics And Main Metrics correlated with the functional requirements	95
6.16	Screenshot of Recommendation to Creation of New Functional Requirements	97
6.17	Screenshot of Recommendation to Change the Priority of Functional Requirements	98
6.18	Recommendation of split a Functional Requirement in two or more New Requirements	99
6.19	Recommendation of Creation of a New Dependency for a Functional Requirement	101
7.1	The Case Study Process from Robert Yin’s guidelines [Yin09]	106
7.2	Screenshot of REQAnalytics with the Requirements Priority Recommendations in Case Study 1	109
7.3	Recommendation to Delete an existing Functional Requirement based on its Web Usage	110
7.4	Case Study 1: Recommendation to Creation of New Functional Requirements	111
7.5	Most used navigation path.	111

7.7	Pages Per Visit with and without recommendations applied using the same period of time, 14 days	112
7.6	Navigation path redesigned.	112
7.8	Screenshot of REQAnalytics with the Requirements Priority Recommendations in Case Study 2	115
7.9	Most Used Navigation Paths related with Requirements in Case Study 2	116
7.10	Traceability Matrix between functional requirements and Web pages and elements in Case Study 2	117
7.11	Entry Features	118
7.12	Exit Features	118
7.13	Split a Functional Requirement in two or more New Requirements	119
7.14	Creation of a New Dependency for a Functional Requirement . . .	119
7.15	Requirements Dependencies Directed Graph of Case Study 2 . . .	120

List of Tables

2.1	An example of a Requirements Traceability Matrix [KS09]	30
3.1	Web Server Log File Types and Content	42
6.1	Web Analytics Tools Comparison	77
6.2	Prioritization of the Requirements	98
7.1	Functional requirements analysed in the Case Study 1	108
7.2	Functional requirements analysed in the Case Study 2	114

Acknowledgments

This thesis is the outcome of my doctoral research carried out at Faculty of Engineering of University of Porto and I would like to thank the collaboration and the support of all those who contributed to its accomplishment.

First and foremost, I wish to express my sincere gratitude to my supervisor, Prof. Ana Paiva, for her continuous scientific support, feedback and suggestions throughout the course of my research and whose expertise, understanding, and patience, added considerably to my graduate experience. Thank you for your valuable insights and comments and for providing everything I needed to my research work.

I would like to endorse my thanks to Prof. Eugénio Oliveira and Prof. Augusto Sousa of the Doctoral Program in Informatics Engineering (ProDEI) for all the support provided.

I would also like to thank all my teachers and colleagues of ProDEI who accompanied me during these years at the PhD course and at different levels contributed to I could do this work.

I also thank all other researchers from the Software Engineering Research Group of the Faculty of Engineering of University of Porto for providing a very good environment for research.

A very special acknowledgement to my parents Manuel and Josefina, my sister Cláudia, my brother Luís and the rest of my family, for your constant and unconditional support that helped and inspired me to complete this thesis.

My final thanks goes to my sweet daughter Carolina and my lovely wife Ana, to whom I dedicate this work.

*—An idea not coupled with action
will never get any bigger than the
brain cell it occupied.*

Arnold H Glasow

1

Introduction

1.1 Problem and Motivation	3
1.2 Research Questions and Goals	5
1.3 Research Strategy	7
1.4 Contributions	9
1.5 Results	9
1.6 Structure of Thesis	10
1.7 Publications	12

In Software Engineering, software quality has become a topic of major concern. Particularly, websites are being used not only for displaying static information but increasingly as core business tools, particularly through web services, intranets and web applications that run as support applications to business development [SS10, TKG07].

The websites and web applications are a type of software that must be available

24 hours a day. Furthermore, they should please the customer and must be maintained and improved to adjust to needed changes through the website lifecycle [MA15].

The main reasons of software project failures are the incorrect requirements elicitation, requirements changes and their uncontrolled evolution during the software project lifetime [Sta95]. Uncontrolled requirement changes cause negative impacts in software development, such as, excessive costs and a system unable to answer to the needs of its stakeholders.

The requirements management allows to maintain stability and agreement among stakeholder's requirements, by means of the analysis of change effect and their monitoring during software lifetime [IWD09]. Software organizations are improving the methods they use to collect, analyze, document, and maintain their requirements in a structured software requirements specification written in natural language. However, a Software Requirements Specification is difficult to keep current, specially when the software project is a website that evolves during its lifetime [Wie99]. Through time, this causes the requirements become outdated and do not reflect the current state of development of websites.

Furthermore, during software lifetime it is difficult to determine what requests of requirements changes should be answered first and the information of traceability between the requirements and the implementation is frequently lost.

Nowadays, there is little support for websites and web applications evolution, despite the evolution of website accounts for major development costs. The evolution of websites involves creation of new requirements, change of existing requirements and changes of the implementation. These activities may come from different stakeholders, such as developers, systems engineers, users and service integrators.

There are methods for measurement, data collection and data analysis of websites and web applications throughout their lifetime. These methods are called Web Analytics. The main objective of Web Analytics is to provide the right direction to online users. This can be done by doing required and impactful changes in the web site [KSK12].

Nowadays, existing Web Analytics tools are able to gather diverse data about the usage of a website. The use of websites generates large amounts of information

that may be used for different purposes like assessment of quality of web-products [SKS14], pattern recognition or to statistical analysis of website data usage [KSK12].

Nevertheless, Web Analytics have some limitations for the maintenance of services. Web Analytics tools have focused on analysis and reporting of business metrics like number visits and traffic sources, which interest is mainly to marketers [PRRS13] and an analysis directed to the improvement is not currently done and this data is disregarded for the improvement of the quality of a web application. The potential of the analysis of this web usage data is yet to be explored [ABB12].

1.1 Problem and Motivation

It is common sense in the field of software development that uncontrolled and outdated software requirements specification leads to mistakes of user and system requirements specifications and also to many project failures [Wie03, KS09, HKL10].

In a service context in which maintenance gains importance, the analysis of the use of the service may also be useful for the maintenance of the requirements as it may, for example, suggest prioritization changes which may have an impact on new updates/changes of software system. For instance, a requirement more used or accessed by the website users may have a higher priority change than other with less usage.

Currently, Web Analytics tools are able to collect diverse data about the usage of a website. However, these tools only generate reports with the navigation statistics, duration of navigation on the website and other metrics that are mostly often used simplistically to see which content with more adherence by users [Ver11]. Either, Web Analytics has focused on analysis and reporting of business metrics of interest mainly to marketers [PRRS13].

Web Analytics tools available [SKS14] do not provide functionalities that enable a more intelligent analysis to suggest improvements to the website, such as suggest new workflows, identify and remove unused features or present more readable and intelligible reports.

The existing software systems continue to be built without meeting user needs. There is a need to quality-oriented approaches to development which involve specifying user and quality requirements and using these requirements to control and evaluate the development process [Fin15, MJZCH13].

In addition, nowadays, the Requirements Management of a website is only done to solve detected problems, satisfy or correct the failures of the software requirements specification [KMA⁺13] not focused in its improvement, and therefore in the improvement of the quality of the website.

Software Quality makes no sense without reference to requirements and requirements management is a prerequisite for quality-oriented development [Fin15]. Furthermore, quality-oriented development is requirements-driven development [MJZCH13]. Requirements management is also required by Capability Maturity Model for software (CMM) developed by the Software Engineering Institute of Carnegie Mellon University and the ISO 9000 series of standards developed by the International Standards Organization (ISO) and most large system customers.

Research in the field of Requirements Management based on data collected of the usage of website is very few [GP14], or not exactly related with the research proposed in this project. For instance, Gao [GLX⁺11] proposed a solution where the evolution of software requirements models is based on the feedback collected. Banerjee [Ban11] presented a methodology to manage requirements based on errors that may occur in the introduction or updating of the software requirements. Ghezzi [GPST14] presents an approach that automates the acquisition of user-interaction requirements through web logs and analyses them by means of probabilistic model checking to identify navigation anomalies and emerging users behaviours.

In a survey related to software requirements specification in model-driven development, Valderas [VP11] demonstrated that few of the existing approaches are specifically defined for the specification of web application requirements. Furthermore, once requirements are specified, there is little support for allowing the systematic or automatic derivation of the conceptual model that properly satisfies the software requirements specification.

There is very few research on Requirements Management based on Web

Analytics and the existing research is also very recent.

Web usage mining is a field with immense potential itself for recommendations not only for users based on their preferences [DM03] but also for systems developers and engineers to help in the website improvement [KSK12]. However, recommendation technologies will only succeed if they deliver high quality recommendations.

Despite of continuous and positive research in the field of Web Analytics, there are still some challenges which researchers need to work upon. The analysis and research done till this moment show that some of the open challenges on this field are:

- It is extracted little knowledge of the web usage data collected with the Web Analytics tools.
- There are many Web Analytics tools with large volumes of data, but they do not give any recommendations to the improvement of the website based on the data collected.
- Stakeholders are often skeptical regarding a new form of automated tool support [MT13]. Therefore, should be delivered high quality recommendations to answer the stakeholders's doubts.
- There is no focus on the improvement of the software requirements specification.
- Available tools support some kind of analysis, however, are not able to perform other kind of analysis. For instance, do not analyse typical paths taken (which may be useful to define or improve workflows); do not produce reports based on the requirements and, therefore, a language closer to the business.

1.2 Research Questions and Goals

Based on the needs captured during exploratory research, this work intends to answer six primary research questions one main research goal and two secondary goals that derived from the research questions:

1.2.1 Research Questions

The six primary research questions that were identified through the preliminary research work and with the information collected are:

- **Q1** *How web usage mining can support requirements change management?*
- **Q2** *How to create traceability links between the functional requirements and the website application?*
- **Q3** *How to analyze web usage data in order to suggest new navigation paths related with the functional requirements?*
- **Q4** *How to generate recommendations to the software requirements specification?*
- **Q5** *What type of recommendations can be suggested to functional requirements?*
- **Q6** *How to provide more readable reports in a language closer to the business?*

1.2.2 Research Goals

The main goal of this research work is to define an approach to improve the maintenance of services and their requirements based on the usage of a website. Using a Web Analytics tool to gather the usage of a website, a recommender system generates recommendations reports that may help the requirements maintenance and increase the quality of the software requirements specification of the website.

To support the main goal, other two secondary goals have been defined: (1) develop a mapping tool that maps the functional requirements of the software requirements specification of a given website with the web pages and elements of the website; and (2) generate more legible and high-level reports of web usage data in a language closer to the business than the reports provided by existing Web Analytics tools.

The research questions are further detailed on Section 5.3 (p. 65) and research goals on Section 5.4 (p. 66).

1.3 Research Strategy

Zelkowitz and Wallace [ZW98] classified research methodologies into scientific, engineering, empirical, and analytical. In the scientific method, a theory is formulated in order to explain a phenomenon. An engineering method aims at formulating a hypothesis and tries to develop and test a proposed solution. An empirical method uses statistical methods to validate a given hypothesis. Finally, the analytical method develops a formal theory. These research methodologies can be applied to science in general, but for the software engineering domain, other specific approaches may be combined.

Zelkowitz and Wallace [ZW98] also proposed some engineering validating methods for validating technology. They are categorized in three categories: (i) Observational methods, (ii) Historical methods, and (iii) Controlled methods.

Observational methods collect data considered relevant during the development of the project. There are four types: project monitoring; case study; assertion; and field study.

Historical methods collect data from projects that have already been completed using existing data. These methods are: literature search; legacy data; lessons learned; and static analysis.

Controlled methods are classical methods of experimental design used in other scientific disciplines and gather information from multiple instances of an observation for statistical validity of the results. There are four types: replicated experiment; synthetic environment experiment; dynamic analysis; and simulation.

Using the methodology mentioned above, and since the scientific area of this research work is in the field of Software Engineering, the work of this research consisted in four distinct phases: Information Gathering; Hypothesis Definition; Approach Development; Approach Evaluation.

- **Information gathering.** This research work started with this initial stage of information gathering and synthesizing. In this phase, the work was directed to collect, study, and synthesize information on the research topics most significant for the problem identified.

The main goals defined for this phase were: (1) overview the main questions related to the research field and their open challenges and issues, and (2)

settle the directions of the research. The topics that have been identified with this research were: Requirements Engineering; Requirements Management; Requirements Management tools; requirements evolution and traceability; Web usage data; Web usage mining approaches; Web Analytics tools; Recommender Systems; Recommender Systems for Software Engineering; and Recommender Systems approaches.

The methods used in this phase were historical research methods like literature search and review. The achieved results from this phase are described in Part I - Background and State of the Art (Chapter 2 (p. 17) - Requirements Engineering, Chapter 3 (p. 39) - Web Usage Mining and Chapter 4 (p. 51) - Recommender Systems).

- **Hypothesis Definition.** With the research carried out and with the information gathered and studied, an hypothesis was formulated Section 5.2 (p. 65). The solution is based on a new approach for Requirements Management and maintenance through recommendations to the requirements specification based on the web usage data of the website.
- **Approach Development.** Based on the hypothesis formulated, the proposed approach was developed. The approach presented in Chapter 6 (p. 71) was implemented through a Recommender System, REQAnalytics that collects information on the usage of a web application, relates that information back to the requirements, and generates reports with recommendations and change suggestions that can increase the quality of that service. The research methods used in this phase were observational ones.
- **Approach Evaluation.** After the proposed solution was developed for the identified problem, observational methods, like case studies, and controlled methods, like replicated experiments, were conducted to validate the solution Chapter 7 (p. 105). The results obtained during this research work were presented in peer-reviewed international conferences and published in peer-reviewed international journals.

1.4 Contributions

The main contributions of this research work are:

- An high-level mapping tool that through a web-based application, maps the functional requirements of the software requirements specification with the implemented functionalities (web pages and elements) of the website. The main goals of this tool are:
 - to reduce the manual work required to maintain the traceability relationship information;
 - maintain traceability relationships between the requirements and web pages and elements of the website;
 - to identify the implemented functional requirements presented in the software requirements specification of a website and therefore trace functional requirement coverage.
- A novel approach to Requirements Management through a Recommender System, REQAnalytics that collects information on the usage of a web application, relates that information back to the requirements, and generates reports with recommendations and change suggestions that can increase the quality of that service;
- Provide more readable reports of web usage data in a language more closer to the business;
- Improve the development of the system, supporting the evolution of the system proposing changes and improvements to the requirements specification.

1.5 Results

This research work has the following results:

- An integrated Recommendation System, REQAnalytics, which provides features for managing functional requirements of services or web applications

and for analyzing information of the usage of such services relating it back to functional requirements. The recommender system is able to provide the follow functionalities:

- Lists of all functional requirements mapped with the web pages and elements of the website;
- Identifies the most used functional requirements, the top entry requirements and the top exit requirements;
- Identifies the most used website navigation paths;
- Identifies all traversed sequences of functional requirements along the web application;
- Automatic generation of the traceability matrix between functional requirements and implementation artifacts;
- Recommendations to the web application Functional Requirements:
 - * Change the priority of a requirement;
 - * Create new requirement;
 - * Delete existing requirement;
 - * Split existing Requirement;
 - * Create new requirement dependency;
 - * Remove requirement dependency.
- Assist in the task of determining the website’s coverage of the requirements through mapping tool developed.
- Provide detailed reports that can effectively give hints/suggestions of how to improve the quality and level of service of websites and web applications. Traditionally, Web Analytics tools offer reports based on the web usage. However, there is a gap in the real utility and comprehensibility of the information gathered.

1.6 Structure of Thesis

This dissertation thesis is logically organized in three parts. The First Part, comprising Chapters 2 to 4, gives a brief background to support the problem

research and presents the proposed solution presented in the Second Part, that is composed by chapters 5 to 6. Finally, the Third Part presents the Validation and summarizes with the Conclusions and Opportunities for Future Research Work. A brief description of each Part and its Chapters is next provided.

Part I: Background and State of the Art

The First Part reviews the most important concepts and issues relevant to the thesis, namely:

- **Chapter 1: Introduction.** Provides a brief introduction to the main research area of this thesis, the problem and the motivation, the main goals, the research strategy, the contributions and results achieved, as well as the structure of the thesis and a list of the publications achieved during the development of this research.
- **Chapter 2: Requirements Management.** Focus on the concepts and terminology related to the field of Requirements Management, the research area of the problem identified. It also gives a state of the art of existing techniques and tools used in the Requirements Management process and in the Requirements Evolution.
- **Chapter 3: Web Usage Mining.** Gives a detailed depiction of the state of the art of Web Mining, particularly Web Usage Mining, that allows for the collection of Web access information for Web pages. It also presents the application of these techniques in the context of Requirements Management addressing the identified research opportunity.
- **Chapter 4: Recommender Systems for Software Engineering.** Discusses the evolution of Recommender Systems. Further presents a state of the art review on the topic of Recommender Systems for Software Engineering, describing existing approaches and identifying the flaws and the open issues.

Part II: Research Approach and Solution

The Second Part states the problem research approach and presents the proposed solution:

- **Chapter 5: Research Problem.** Considers the state of the art, formulates the thesis statement, overviews the solution proposed, the methodology that was followed and draws the validation strategy.
- **Chapter 6: REQAnalytics.** Presents the proposed solution describing a novel approach for Requirements Management through a Recommender System that generates recommendations to the software requirements specification based on the web usage data of the website.

Part III: Validation and Conclusions

The Third and final Part presents two experimental evaluations for the validation of the thesis, and presents the conclusions of this research work and set future directions:

- **Chapter 7: Validation.** Gives two thorough experimental evaluations of different type of websites, assessing the efficiency of the recommendations to the requirements specification presented by the proposed recommender system and discussing the results achieved with REQAnalytics.
- **Chapter 8: Conclusions and Future Work.** Discusses the research work done, summarizing the contributions made and points opportunities for future work directions.

1.7 Publications

Parts of this thesis dissertation have already been published in international conferences and journals during the author's Ph.D. research work. A list of those is given next chronologically ordered:

- **A Requirements-to-Implementation Mapping Tool for Requirements Traceability**, a state of the art on Requirements traceability, with an approach that supports the mapping of functional requirements with the

web pages and elements of a web site to help maintaining requirements traceability information updated and, ultimately, increasing the efficiency of requirements change management process itself that may contribute to the overall quality of the service provided.

This work was presented in the **8th International Conference on Computer Science and Information Technology (ICCSIT 2015)**, Amsterdam, Netherlands, and a revised version was selected to publication in the **Journal of Software** [GP16a].

- **REQAnalytics: A Recommender System for Requirements Maintenance**, a preliminary approach of REQAnalytics, a recommender system that collects information on the usage of a web application, relates that information back to the requirements, and generates reports with recommendations and change suggestions to the requirements specification baseline like features that can be removed, change the priority of requirements and presents the relationship between requirements and the proposed changes helping to maintain the software requirements specification updated and useful that can increase the quality of that service.

This paper has been published in the **International Journal of Software Engineering and Its Applications** [GP16b].

- **An Automated Approach for Requirement Specification Maintenance**, presents an experimental evaluation of a case study based on an online newspaper website using an approach through a recommender system that collects the information about the usage of a website using a Web Analytics tools and generate recommendations reports that may help the requirements maintenance and increase the quality of the software requirements specification of the website.

This paper was accepted and will be presented in the **4th World Conference on Information Systems and Technologies**, Recife, Brazil and will be published in the **Journal of Advances in Intelligent Systems and Computing Series**.

Part I

Background and State of the Art

—*What gets measured gets managed.*

Peter Drucker

2

Requirements Engineering

2.1 Introduction	18
2.2 Requirements Engineering	22
2.3 Requirements Management Tools	30
2.4 Discussion	37
2.5 Summary	37

The main goal of this work is to improve the maintenance of services and their requirements based on the usage of a website. The purpose of this chapter is therefore to introduce the reader to fundamental concepts related to Requirements Engineering (RE) and Requirements Management (RM).

2.1 Introduction

Today our society is increasingly dependent on software, and its quality is the key to its success. In software engineering, software quality has become a topic of major concern. As software provides more support to the organizations for achieving their goals, it becomes extremely important for the organizations to be competitive in their business, and consequently its quality and utility should be a major concern.

Requirements engineering is related to the process of eliciting individual stakeholder requirements and needs and developing them into detailed, agreed requirements documented and specified in such a way that they can serve as the basis for all other system development activities [Poh10].

The processes used for Requirements Engineering are used depending on the application field and on the organization developing the requirements. However, there are some general activities [Low99] common to all processes: Requirements elicitation; Requirements specification; Requirements validation; Requirements management. Requirements Engineering is an iterative process in which the activities are all interleaved.

2.1.1 Requirement Definition

A requirement is defined as a feature, property, capability or behaviour that must be fulfilled or met by a system or software application. The requirements must be quantifiable, relevant, verifiable, traceable and detailed.

The IEEE [IEE10] defines a requirement as:

1. a condition or capability needed by a user to solve a problem or achieve an objective
2. a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document
3. a documented representation of a condition or capability as in definition 1 or 2

4. a condition or capability that must be met or possessed by a system, product, service, result, or component to satisfy a contract, standard, specification, or other formally imposed document. Requirements include the quantified and documented needs, wants, and expectations of the sponsor, customer, and other stakeholders.

In the field of Software development, requirements can be also called functional specifications, which is the documentation that describes the requested behaviour of an engineering system.

2.1.2 Taxonomies

Considering the different aspects of the products lifecycle, Hooks and Farry [HF01] proposed an extensive requirements taxonomy. Dong [Don02] has also classified as a subject-based classification.

This taxonomy included the following type of requirements:

- Functional requirements
- Performance requirements
- Physical requirements
- Aesthetic requirements
- Manufacturing requirements
- Assembly requirements
- Reliability requirements
- Safety requirements
- Maintainability and serviceability requirements
- Packaging requirements
- Transportation requirements

2.1.3 Requirements Types

More recently and according to requirements engineering standards [Som07, Wie03], requirements are commonly divided into three types: (i) Functional Requirements; (ii) Quality Requirements; and (iii) Constraints.

Functional Requirements

Functional Requirements define the functionality the system shall provide to its users [Poh10].

Sommerville [Som07] defines functional requirement as:

These are statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do. These requirements depend on the type of software being developed, the expected users of the software and the general approach taken by the organisation when writing requirements. When expressed as user requirements, the requirements are usually described in a fairly abstract way.

According to Pohl [Poh10], Functional requirements describe functionalities that the system shall provide to its users.

Quality Requirements

A quality requirement is a property of the system to be developed, e.g. the performance of the system, is reliability, or its stability. Pohl [Poh10] defined as follows:

A quality requirement defines a quality property of the entire system or of a system component, service, or function.

There are different types of quality requirements. Wiegers [Wie03] identified the following quality requirements:

- **Availability:** refers to the percentage of time during which the system is actually available for use and fully operational
- **Efficiency:** is a measure of how well the system uses hardware resources such as processor time, memory, or communication bandwidth

- **Flexibility:** indicates how much effort is needed to extend the system with new capabilities
- **Integrity:** denotes how well the system is protected against unauthorised access, violations of data privacy, information loss, and infections through maleficent software
- **Interoperability:** indicates how easily the system can exchange data or services with other systems
- **Reliability:** is the probability of the system executing without failure for a specific period of time
- **Robustness:** is the degree to which a system or component continues to function correctly when confronted with invalid inputs, defects in connected systems or components, or unexpected operating conditions
- **Usability:** measures the effort the user requires to prepare input for, operate, and interpret the output of the system
- **Maintainability:** indicates how easy it is to correct a defect or make a change in the system
- **Portability:** relates to the effort it takes to migrate a system or component from one operating environment to another
- **Reusability:** indicates the extent to which a component can be used in systems other than the one for which it was initially developed
- **Testability:** refers to the ease with which the software components or integrated system can be tested to find defects

Constraints

Constraints are a type of requirements used to either restrict the development process or the properties of the system to be developed. Constraints can typically not be changed by stakeholders involved in the RE process. Robertson and Robertson [RR06] defined as follows:

A Constraint is an organizational or technological requirement that restricts the way in which the system shall be developed.

2.1.4 Other Requirements Taxonomies

Sommerville and Kotonya [SK98] have traditionally distinguished between functional and non-functional requirements). They define non-functional requirements as "the overall qualities and attributes of the resulting system including any constraints on interfaces, quality, resources or time-scales"

Additionally, Sommerville and Kotonya [SK98] state that non-functional requirements include performance, interface, operational, resource, safety, portability, quality and reliability or maintainability requirements.

This approach is however not clearly accepted in literature and several authors disagree with these classification. For instance, Pohl [Poh10] disagree with these classifications and strongly recommends not using the term non-functional requirement, due to the fact that non-functional requirements either are under-specified functional requirements or quality requirements as defined above.

2.2 Requirements Engineering

Requirements engineering (RE) is related to the process of eliciting individual stakeholder requirements and needs and developing them into detailed, agreed requirements documented and specified in such a way that they can serve as the basis for all other system development activities [Poh10]. Traditionally is carried out in the beginning of the system development lifecycle [Roy90].

Requirements Engineering, also called requirements analysis is the process of determining the necessary conditions for a new or modified software product.

Zave [Zav97] states one clear description of RE:

Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families.

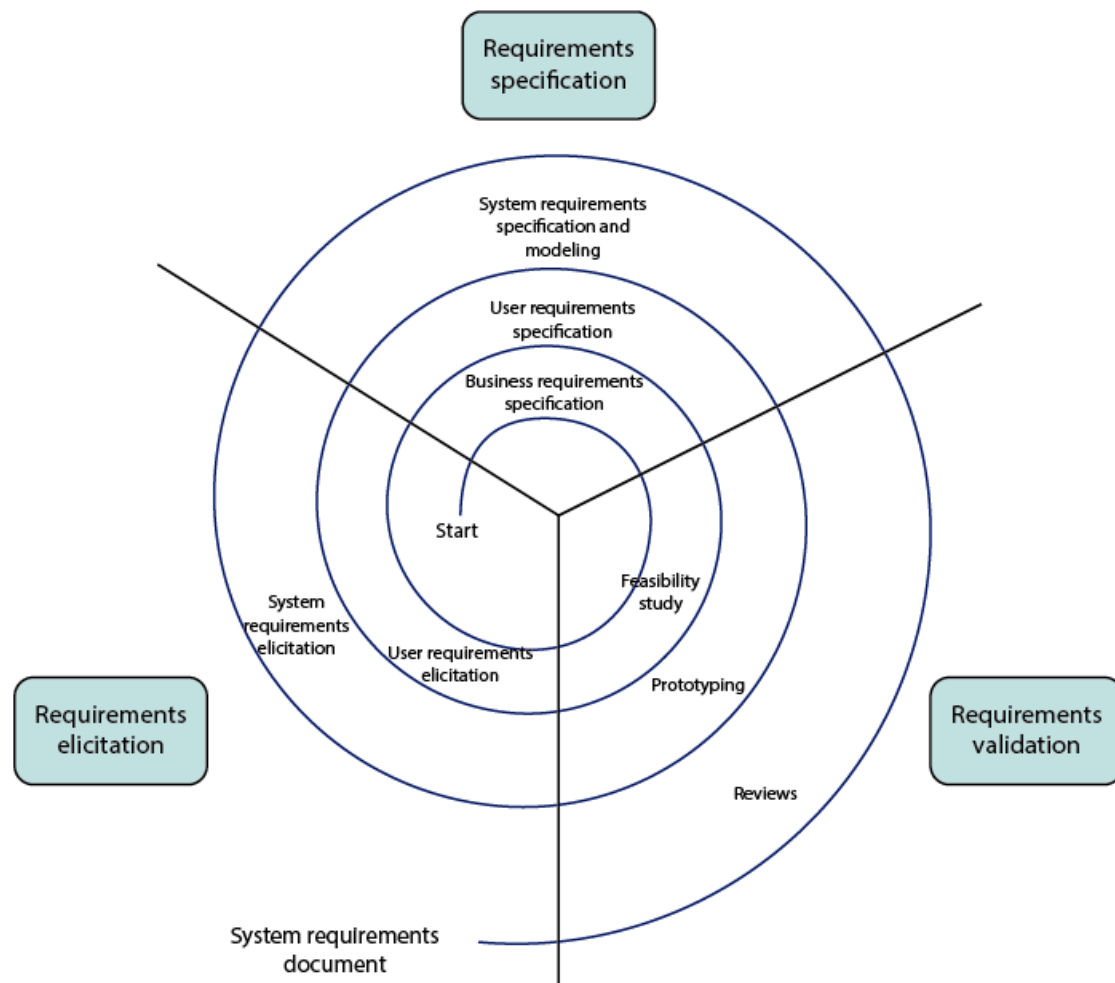


Figure 2.1: Requirements engineering processes

The processes used for Requirements Engineering are used depending on the application field and on the organization developing the requirements. However, there are some general activities [Low99] common to all processes:

- Requirements elicitation
- Requirements specification
- Requirements validation
- Requirements management

Figure 2.1 shows a spiral model view of the requirements engineering process. Requirements Engineering is an iterative process in which the activities are all interleaved.

2.2.1 Requirements Elicitation

The Requirements elicitation, sometimes also called Requirements discovery is the process of discovering, reviewing, documenting, and understanding the user's needs and constraints for the system. Involves working with the customers to understand the application domain and may involve other people like end-users, managers, domain experts. These are called the stakeholders.

2.2.2 Requirements Specification

Requirements specification is the process of documenting the user's needs and constraints clearly and precisely. On this process it is necessary to fully describe what the application will do and how it is expected to perform.

2.2.3 Requirements Validation

Requirements validation is an iterative process of ensuring that the system requirements are complete, correct, consistent, and clear. Is also necessary to verify if the requirements statement themselves are complete, correct, feasible, necessary, prioritized, unambiguous and verifiable. There are some validation techniques like requirement reviews, prototyping and test-case generation.

2.2.4 Requirements Management

Requirements management can be described as the process of managing changing requirements during the requirements engineering process and system development [Somo4].

During the development of a system and also after it being delivered to the customer, new requirements emerge and it is necessary to keep track of this new requirements. The process of software requirements management practices is believed to be one of the first process improvements steps that a software development organisation should take [EMMo1, DCVPo3].

In the last years, Requirements Management is increasingly recognised as crucial [NEoo], due to the needing of writing requirements readable and traceable, in order to manage their evolution over time. The requirements evolution

consists in the changes made to the requirements after the initial deploy of the requirements specification document [AP01]. Requirement changes may consist in additions, omissions or change and can occur in any process, elicitation, analysis, specification or validation. A software requirements specification evolves when a system change is performed, its behavior changes and it is necessary to change some of the requirements initially specified. Nowadays, the task of requirements management is considered extremely important to high quality software and to project success [ED, LvDNdZ04, SAW13, LQF10], however the requirements management is often ignored.

Figure 2.2 shows the major activities that are related to Requirements Management that have been identified by Wiegers [Wie03].

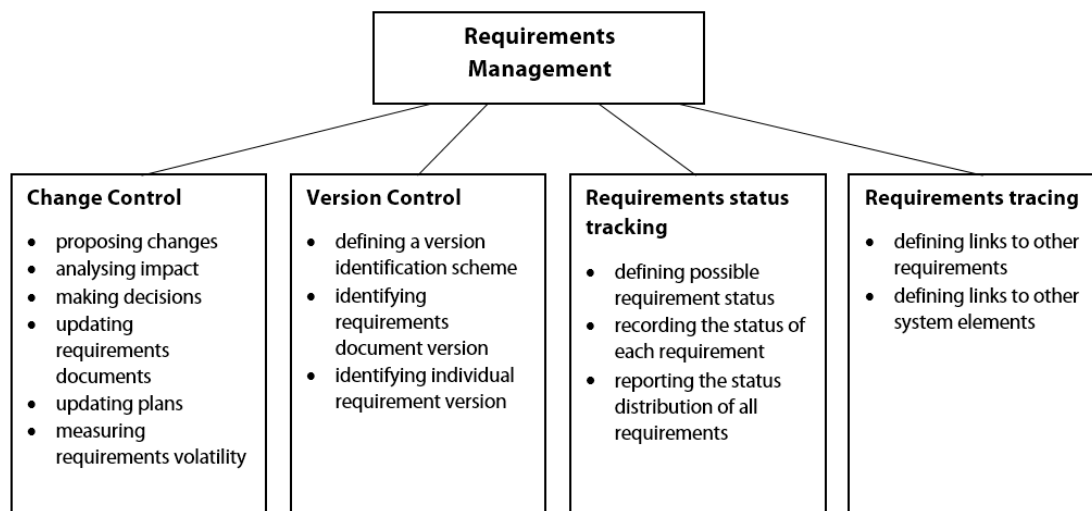


Figure 2.2: Major RM activities [Wie03]

Sommerville and Kotonya specified these activities in three main areas related to requirements management [SK98]: (i) Change management, (ii) Requirements attributes, (iii) Requirements traceability.

Change Management

Change management describes the procedures and tasks needed to be followed when analysing a requirements change. Sommerville [Som04] proposed a workflow process, shown in Figure 2.3, to change management. Starts with problem analysis in which requirements are discussed and changes are proposed. Then

change requests are analysed in order to assess their impact. Finally, the change is implemented, i.e., the documents are changed to reflect the change.

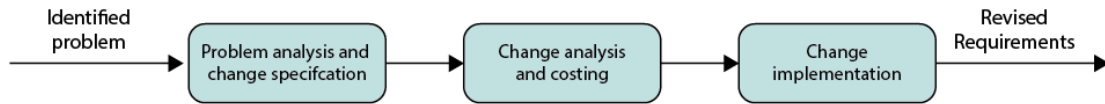


Figure 2.3: Change management workflow process

Requirement errors are the largest type of errors typically found in a project, and it is estimated that 56 percent of the discovered errors are related to the requirements specification [HF01, DSo8]. Reducing requirement errors can be the single most effective action developers can take to improve project outcomes and identifying omitted requirements and finding errors during the requirements stage, as opposed to later stages of the life cycle, provides great leverage and cost savings [Wil11].

Traditionally, most requirements engineering tasks, including the requirements elicitation, are carried out in the beginning of the system development lifecycle [Roy90]. However, when the software application is a service provided continually over time, as are web applications and services, requirements may change or new requirements may come up for several reasons, for instance, new laws, new needs, etc.

Requirements Attributes

Requirements attributes are information necessary to better describe and interpret the requirements. This include the requirement classification (e.g. functional or non-functional, priority) and the verification method or acceptance test plan. It can also include other information like the source of the requirement, change history and the unambiguously requirement ID.

Requirements Traceability

Requirements traceability consist in the recovering the source of requirements and is used to perform impact analysis of a requirement change. A requirement

should be traceable backwards to its origin (e.g. use cases, test cases, source code) and forwards into requirements and design entities that satisfy it.

2.2.5 Requirements Evolution

The requirements evolution consists in the changes made to the requirements after the initial deploy of the requirements specification document [APo1].

Requirement changes may consist in additions, omissions or change [SOSA99] and can occur in any process, elicitation, analysis, specification or validation.

A software requirements specification evolves when a system adaption is performed, its behaviour changes and it is necessary to change some of the requirements initially specified.

2.2.6 Traceability

Keep traceability information between web pages and elements of the website and requirements may be useful to evaluate the impact of change requests and help maintenance activities. Software traceability has been increasingly recognized as an important quality of a well-engineered software system [RJ01]. It is defined by the Center of Excellence for Software and Systems Traceability (CoEST) as the ability to interrelate any uniquely identifiable software engineering artifact to any other, maintain required links over time, and use the resulting network to answer questions of both the software product and its development process [coe].

Software traceability is recognized as a critical success factor in software development [DP98] and has been recognized as an important quality of a well-engineered software system [GF94]. Requirements traceability has been demonstrated to provide many benefits to organizations that make proper use of traceability techniques [KS09].

The USA Food and Drug Administration (FDA) states that traceability analysis must be used to verify that the software design implements the specified software requirements, that all aspects of the design are traceable to software requirements, and that all code is linked to established specifications and test procedures [F0005]. In another report, "Critical Code: Software Producibility for Defense", the Committee for Advancing Software Intensive Systems Producibility

of the U.S. Department of Defense identified requirements traceability as one of the seven technology areas on which research should be targeted in order to assure the safe and correct operation of current and future software intensive systems [Com11].

Requirements traceability has been defined by Gotel and Finkelstein [GF94] as the ability to describe and follow the life of a requirement, in both a forward and backward direction (i.e. from its origins, through its development and specification, to its subsequent deployment and use, and through periods of ongoing refinement and iteration in any of these phases). These traceability chains are shown in Figure 2.4.

Despite the benefits resulting from the use of requirements traceability techniques, the poor tool support is perhaps the biggest challenge to the implementation of traceability [Wil11] [KS09]. Furthermore, the use of requirements traceability tools is just used for about 50% of the software engineering industry [LM06]. One of the main reasons for this low rate is because in the existing requirement management tools, exists poor support for traceability and are inadequate for the needs of the software engineering industry [Geo].

Kannenbergh and Saiedian [KS09] states that creating cost-effective requirements traceability tools that improve upon the design and feature set of currently available tools would serve to greatly improve the practice of traceability in the software engineering industry.

In addition, Di and Zhang [DZ09] stated that the recovering traceability links between requirement and other artifacts is not yet well investigated and there is a stringent need for an automatic tracing method to trace high-level requirements to other software artifacts which are expressed in natural language and may evolve autonomously. Moreover, these techniques are used mainly between requirements and software test cases and the majority of these tools are unable to automatically generate and maintain traceability relationships [HDO03, TWF⁺15].

Forward traceability enables to trace the requirements sources to their resulting product requirement(s) to ensure the completeness of the product requirement specification. Tracing each unique product requirement forward into the design that implements that requirement, the code that implements that design and the tests that validate that requirement and so on. The objective is to ensure that

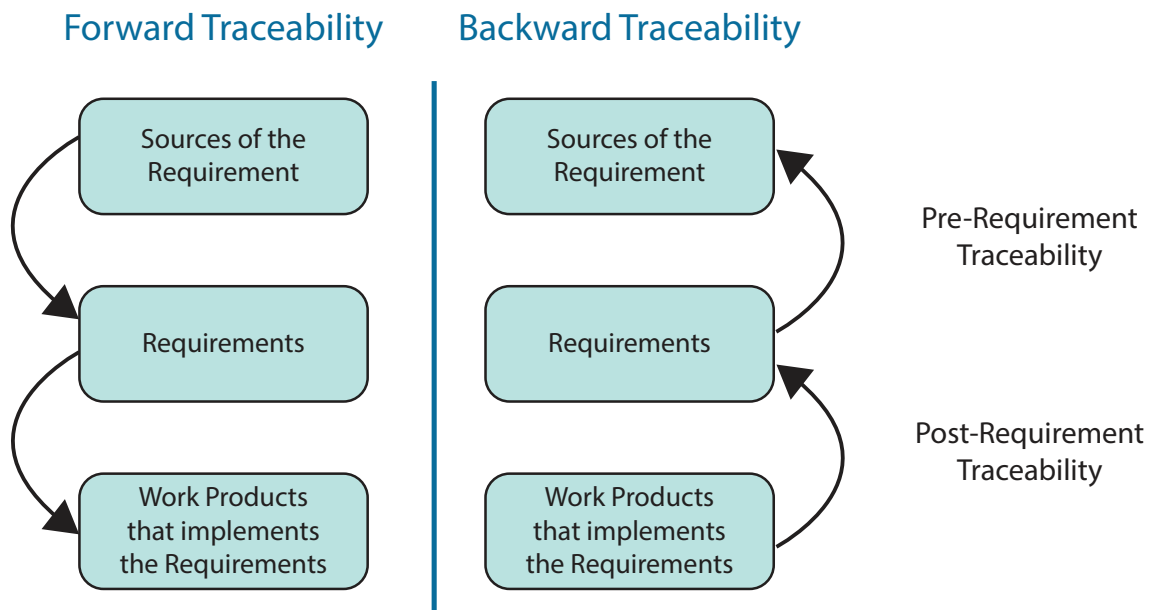


Figure 2.4: Bidirectional - Forward and Backward Traceability [Weso6]

each requirement is implemented in the product and that each requirement is thoroughly tested. [Weso6].

Backward Tracing is related to trace each unique work product (e.g., design element, object/class, code unit, test) back to its associated requirement. Backward traceability can verify that the requirements have been kept current with the design, code, and tests [Weso6].

Requirements Traceability Matrix

The most used method for tracing requirements to their outcomes is a Requirements Traceability Matrix (RTM). A RTM is defined as a table that illustrates logical links between individual functional requirements and other system artifacts [Wieo3]. However, manual traceability methods are not suitable for the needs of the software engineering industry since traceability links that need to be captured grow exponentially with the size and complexity of the software system [CHCCo3]. Traceability analysis is related to the process of tracking forward or backward through a network of interrelationships between components of a system and their documentation.

An example Requirements Traceability Matrix is shown in Table 2.1.

Requirements Traceability is also an important methodology during the maintenance phase of an application or web application. The initially defined require-

Table 2.1: An example of a Requirements Traceability Matrix [KS09]

System Requirement	Software Requirement	Design Element	Code Module	Test Case
A005-00150-80-00505	005-00150-80-00112	Airspeed Calculation	Calculate_airspeed()	Tc_103.doc
A005-00150-80-00506	005-00150-80-00234	Airspeed Display	Display_airspeed()	Tc_125.doc

ments in the software requirements specification often change during the lifecycle of the project and it is very important to assess the impact of these changes. Traceability allows to determine what requirement, test cases or other artifacts need to be changed and can also determinate the costs and risks associated with that change.

Pohl [Poh96] states that a traceability approach should provide answers to the following questions: what traceability information should be captured, how traceability information should be captured and how traceability information should be stored.

Sherba [SA03] added another question that a traceability approach should also answer: how traceability relations are going to be viewed and queried.

2.3 Requirements Management Tools

There are several research and commercial tools that provide capabilities for documenting requirements, managing their changes and support traceability between artifacts [NE00, CNA⁺11, HKL10, KKKCo8]. Requirements management tools are increasingly used to ease the Requirements Engineering processes and allow for more systematic and formalized maintenance of requirements, change management and traceability.

Figure 2.5 shows a table from a deep survey [CNA⁺11] with the main software solutions for Requirements Engineering, detailing its features, price, and a score to classify the best feature for each solution .

Due to the large number of RE tools, we discussed here only three, which are recognized as noteworthy based on usage and market presence

2.3.1 IBM Rational Doors

DOORS is a requirements management application for optimizing requirements communication, collaboration and verification throughout your organization and supply chain. Figure 2.6 shows a screenshot of the tool.

It provides several features of Requirements Management, Traceability and Test tracking. Doors also provides visualisation of documents as hierarchies and its extension language enables a wide range of supporting tools to be built, and many are provided as menu commands and examples.

2.3.2 Rational RequisitePro

RequisitePro is specially designed for project teams who want to manage their application software requirements, write good use cases, improve traceability, strengthen collaboration, reduce project risk, and increase quality.

Rational RequisitePro is an easy to use requirements management tool that lets teams share their requirements using familiar document-based methods while leveraging database-enabled capabilities such as requirements traceability and impact analysis. The result is better communication and management of requirements with the increased likelihood of completing projects on time. Successful projects start with requirements management - the more effective the maintenance of requirements, the greater the resulting quality and customer satisfaction.

2.3.3 CaliberRM by Borland

CaliberRM is a complete requirements solution which ensures compliance and alignment of development to the business needs. It facilitates stakeholder collaboration, rich visualization, robust management, and traceability of requirements to Agile delivery plans.

Some of the main features of this tool are:

- Requirements management. Manage the requirements with version control, traceability, impact analysis, and requirement reuse. Also, provides the traceability to model internal relationships between requirements, as well as external relationships with regulatory controls. This traceability supports impact analysis, so you can understand when requirements change and how those changes impact the project.
- Requirements visualization. Allows to visualize requirements as interactive, testable user flows to increase clarity and precision and automatically generate test cases from defined scenarios and collect feedback in context to ensure that requirements are defined completely and unambiguously.
- Traceability and real-time impact analysis. Provides automatic generation and reporting of customized trace tables. The built-in control detects orphans and highlights suspect traces.
- Collaboration and review. Facilitate collaboration across business stakeholders with Caliber Review. See related attributes, traces and discussions, to ensure stakeholder participation and feedback. Allows to create filters to show only items which require review/approval, and approve baselines with electronic signatures.

2.3.4 Requirements Management Approaches

In other traceability approaches, there is some work related to different tools and methodologies for requirements traceability. Saiedian [SKM11] stated that existing traceability tools focus primarily on requirements traceability or traceability among the various artifacts of a software product, however, there is still an open issue in end-to-end traceability.

For instance, Sherba and Anderson [SA03] proposed a traceability, which has an evolution service that analyzes the changes to a set of relationships over time. It analyzes existing links in different versions without evolving the links themselves. Huffman Hayes et al. [HDS06] developed a traceability tool, called RETRO (Requirements Tracing On-target [Weso6]), to trace requirements. RETRO implements both VSM and LSI for determining requirements similarity.

Egyed and Grunbacher [EG02] presented a tool-supported technique easing trace acquisition by generating trace information automatically.

Neumuller and Grunbacher [NG06] developed a traceability environment that was introduced in a very small software company where they have found out that comparably simple automation techniques were surprisingly effective. Cuddeback et al. [CDH10] presented a framework for the study of analyst interaction with artifacts generated automatically during the tracing.

Neumuller and Grunbacher [NG06] developed a traceability environment and introduced in a very small software company where they have found that comparably simple automation techniques are surprisingly effective. Cuddeback et al. [CHGH⁺14] presented a framework for the study of analyst interaction with artifacts generated automatically during the tracing.

Saiedian [SKM11] stated that existing traceability tools focus primarily on requirements traceability or traceability among the various artifacts of a software product, however, there is still an open issue in end-to-end traceability. For instance, Sherba and Anderson [SA03] proposed a traceability, which has an evolution service that analyzes the changes to a set of relationships over time. It analyzes existing links in different versions without evolving the links themselves. Huffman Hayes et al. [HDS06] developed a traceability tool, called RETRO (Requirements Tracing On-target [20]), to trace requirements. RETRO implements both VSM and LSI for determining requirements similarity. Egyed and Grunbacher [EG02] presented a tool-supported technique easing trace acquisition by generating trace information automatically.

Gotel and Finkelstein [GF94] detected that traceability methods were preferred in the industry due to shortcomings in available traceability tools. However, this problem still exists today because manual traceability methods are still preferred by a significant percentage of software organizations [KS09]. Some research has also been done to investigate the methods to recovery traceability links between design and implementation.

For instance, Egyed and Grunbacher [EG02] proposed a method to recover traceability links between requirements and Java programs by monitoring the Java programs to record the usage of the program classes when scenarios are executed. Despite this, the majority of the related techniques are used mainly between

requirements and software test cases and the majority of these tools are unable to automatically generate and maintain traceability relationships [HDO03].

Li and Walid [LM12] found that matrices and graphs were preferred to support management tasks, while hyperlinks were preferred to support implementation and testing tasks. Matrices were found appropriate to gain an overview of the traceability, while graphs were found suited to navigating the resulting traces.

Tool	Vendor	Elicitation	Analysis	Specification	Modeling	Verification & validation	Management	Traceability	Other tool capabilities	Global	Price range (single seat)
Acclaro DFSS	Axiomatic Design Solutions	+	++	+	n/a	++	n/a	+	n/a	n/a	SSSS
Aligned Elements	Aligned	+	++	+	++	++	0	+	-	+	SSSS
Avenqo PEP	Avenqo, Germany	++	++	+	--	+	0	++	0	+	SSS
Blueprint	Blueprint Software Systems	+	++	+	+	n/a	+	++	+	n/a	SSSS
Bright Green Projects	Bright Green	++	++	+	++	0	+	+	+	+	n/a; ◦
Caliber RM	Micro Focus	++	+	+	-	n/a	++	+	+	n/a	n/a
Cameo Requirements+	No Magic	++	+	0	0	+	-	0	0	0	SSS
CASE Spec	Goda Software	n/a	n/a	n/a	n/a	++	0	++	++	n/a	n/a
Cognition Cockpit	Cognition	++	++	++	++	++	++	++	++	++	SSSS
Cradle	3SL	++	++	++	++	++	++	++	++	++	SSSS
GMARC	Computer System Architects	++	++	++	+	++	+	++	+	++	SSSS
inteGREAT	eDev technologies	++	+	++	++	++	+	++	++	++	SSSS
IRqA	Visure Solutions	n/a	n/a	n/a	n/a	++	++	++	+	n/a	n/a
jUCMNav	jUCMNav	-	-	-	+	n/a	n/a	n/a	n/a	n/a	◦
Leap SE	Leap Systems	-	-	-	-	-	0	0	0	-	SS
MacA&D / WinA&D	Excel Software	+	n/a	+	+	n/a	-	+	n/a	n/a	SSSS
MKS Integrity	MKS	++	++	++	0	++	++	+	++	++	SSSS
PACE	ViewSet	++	++	++	++	++	++	+	++	++	SSSS
Polarion Requirements	Polarion Software	++	++	+	+	++	++	++	++	++	SSS
Psoda	Psoda	++	++	+	+	++	++	+	++	+	S
QFDcapture	International TechneGroup	-	0	0	--	-	--	-	-	-	SSSS
QPack	Orcanos	++	+	++	n/a	++	++	++	+	n/a	SSS
RaQuest	SparxSystems Japan	+	0	+	+	-	+	+	0	+	SS
Rational Doors	IBM Rational	+	+	++	+	n/a	n/a	n/a	n/a	n/a	SSSS
ReqMan	RequirementOne	++	++	+	+	++	+	+	+	+	•
Reqtify & Requirement Central	Dassault Systemes	++	++	+	++	++	++	++	++	++	SSSS
Rational Requirements Composer	IBM Rational	+	+	+	0	n/a	0	+	+	n/a	SSSS
RTIME	QAvantage	++	+	+	0	++	+	+	+	+	SS
Rational RequisitePro	IBM Rational	n/a	+	+	-	n/a	0	+	n/a	n/a	SSSS
RMTrak	Prometeo Technologies	0	-	-	--	+	-	0	-	-	SS
Rommana	Rommana Software	+	+	0	0	n/a	0	n/a	n/a	n/a	SSS
SpiraTeam	Inflectra	++	0	0	--	+	0	0	++	0	S
TestTrack RM	Seapine Software	+	++	+	-	+	+	0	+	+	n/a
TopTeam Analyst	TechnoSolutions	+	0	+	+	0	++	++	+	+	SSSS
TraceCloud	TraceCloud	+	+	++	+	++	++	+	++	+	S
TrackStudio	TrackStudio	+	++	-	-	+	0	0	0	0	S; ◦
VisibleThread On-premise/On-demand	VisibleThread	++	0	++	+	n/a	n/a	n/a	n/a	n/a	SSSS

* For the scores, ++ = very high, + = high, 0 = medium, - = low, and -- = very low. For prices, SSSS > \$1,000, SSS = \$501-\$1,000, SS = \$100-\$500, and S < \$100, free = *, free version available with limitations = ◦, n/a = not applicable.

Figure 2.5: Requirement Engineering tools' scores and prices [CNA⁺11]

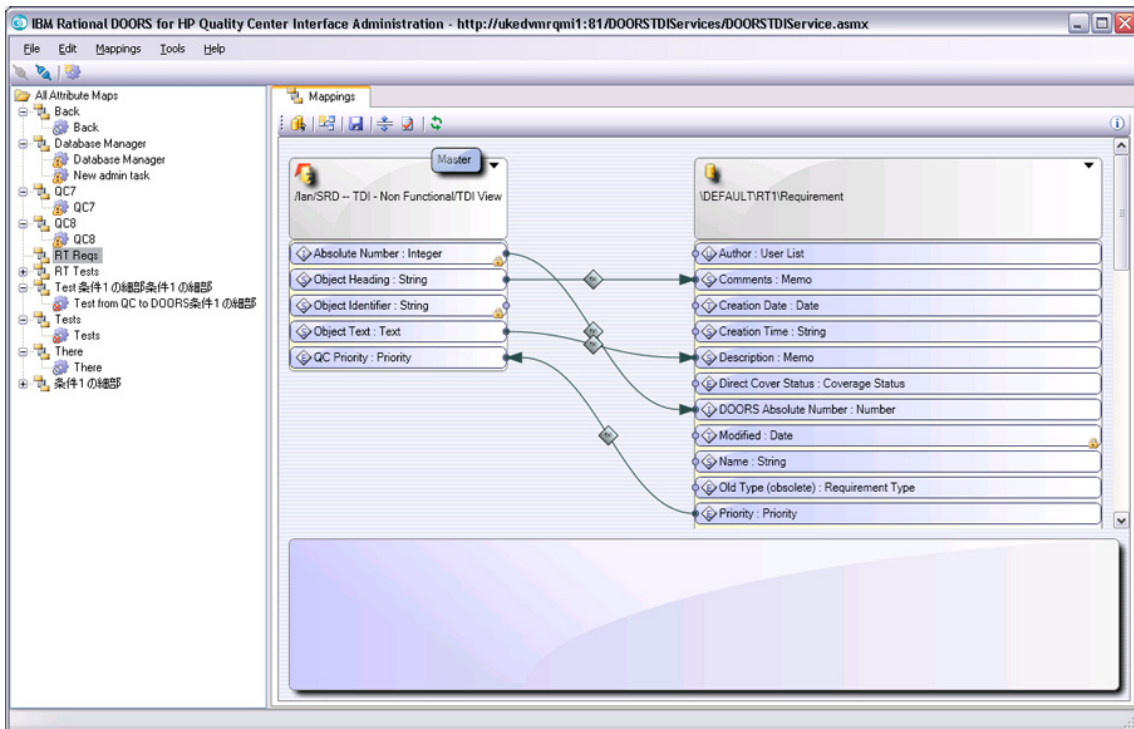


Figure 2.6: Screenshot of IBM Rational Doors

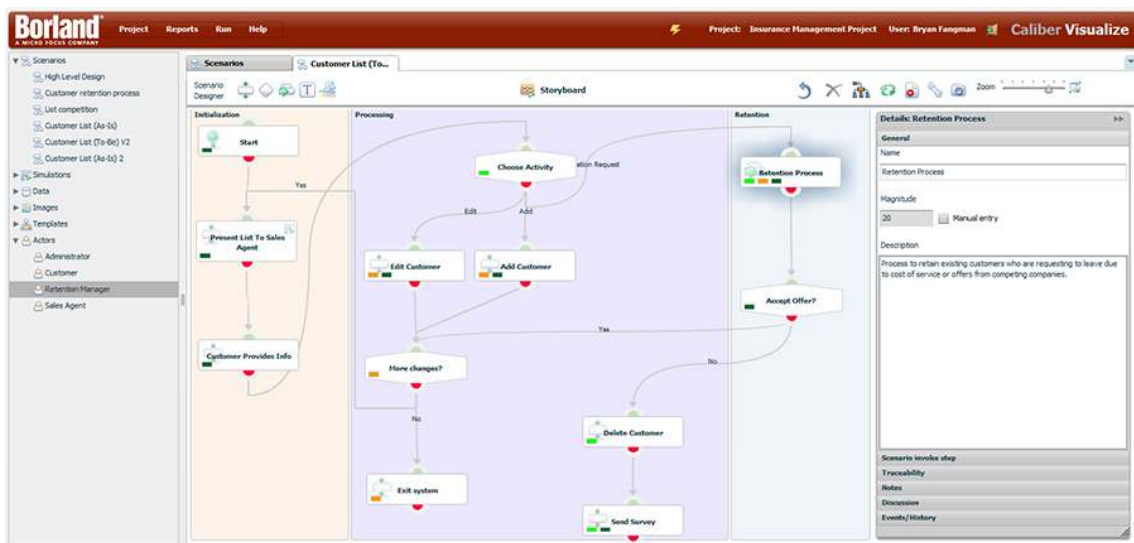


Figure 2.7: Screenshot of CaliberRM

2.4 Discussion

Nowadays, the Requirements Management is only done to solve detected problems, satisfy or correct the failures of the software requirements specification [KMA⁺13] not focused in its improvement, and therefore in the improvement of the quality of the website they describe.

It is common sense in the field of software development that uncontrolled and outdated software requirements specification leads to many project failures. So, in order to increase the lifetime of a service, it is important to deal carefully with requirements managements issues to maintain requirements updated and to prioritize change requests. Otherwise, the requirements' documents get outdated becoming useless. In this context, management of change requests and the analysis of their impact may be complex and extremely difficult.

There are several research and commercial tools, such as DOORS and Rational RequisitePro by IBM, and CaliberRM by Borland available that provide capabilities for documenting requirements, managing their changes and support traceability between artifacts. In other traceability approaches, there is some work related to different tools and methodologies for requirements traceability. Poor tool support is perhaps the biggest challenge to the implementation of traceability [KS09].

2.5 Summary

This chapter presented a review of prior research contributions around the topic of requirements engineering and requirements management in particular and have also included various perspectives arising from literature, such as requirements evolution, and requirements traceability.

The Requirements Management tools market has been developing fast in the latest years. Classic tools like IBM Rational Doors and CaliberRM that used to dominate the market are increasingly complex and getting difficult to use, which leads to some developers opt for alternative solutions.

Moreover, many expensive tools are not fully integrated with tools of other vendors, such as for modelling or traceability. This leads to the arising of new

requirements management tools that introduced interesting capabilities, especially for collaboration and new applications of requirements.

The existing Requirements Engineering tools primarily support the definition and cataloging of requirements but fail to provide additional information such as similarity of requirements, dependencies between requirements, and quality status of requirements [FRST15].

—The goal is to turn data into information, and information into insight.

Carly Fiorina

3

Web Usage Mining

3.1	Web Mining	40
3.2	Web Usage Mining	41
3.3	Web Usage Mining Approaches	44
3.4	Web Analytics	45
3.5	Discussion	47
3.6	Summary	49

Today, the World Wide Web is used by billion of people all over the world. This expansion has led to a large amount of data that exceeds 10 billions pages, or more than six terabytes of data [HK12]. Everyday millions of pages are created, adding gigabytes of new data to the Web. The use of websites generates large amounts of information (e.g. number of visits, duration of visit, page views, click path, exit rate) that may be used for different purposes like assessment of quality of web-products (i.e. websites, web videos) [SKS14] or to statistical analysis of

website data usage [KSK12].

This kind of data, however, is heterogeneous, semi structured or unstructured in contrast to the standard data that the common data mining methods have to deal with [ZSo8].

Researchers are increasingly studying new methods to manage and analyse web data in order to structure, standardize and organize it. Therefore, new mining algorithms were created to better answer the demands of web data. Then, Web Mining (or Web Data Mining) can be described as the whole of data mining and related techniques used to automatically discover an extract information from web documents and services [CL96, KBoo]. Web data can be in several formats (e.g. HTML, XML, Javascript, plain text) and can be stored in different servers just for a single website.

3.1 Web Mining

Web Mining can be categorized to three areas based on objective of information or knowledge intended to extract [KBoo] as shown if Figure 3.1:

- **Web Content Mining:** describes the discovery of useful information or knowledge from web documents. The web content data is considerably heterogeneous and unstructured and consists of several type of data like text or multimedia like image, audio or video.

In web content mining there is also two groups of researching: web page content mining, that is related to retrieval of information available on the web; search result mining, that refers to different approaches of indexing data for an easier knowledge discovery.

- **Web Structure Mining:** refers to analysing and discovering the model of the link structure of the Web. Throughout these methods it is possible to categorize web pages and extract information like the similarity and relationship between them, using the hyperlink structure.

These techniques and methods can be also used to redesign the website that leads [SS10] to an improvement of the quality of a website.

- Web Usage Mining: Collects and analyse the data generated through web-sites visits and transactions.

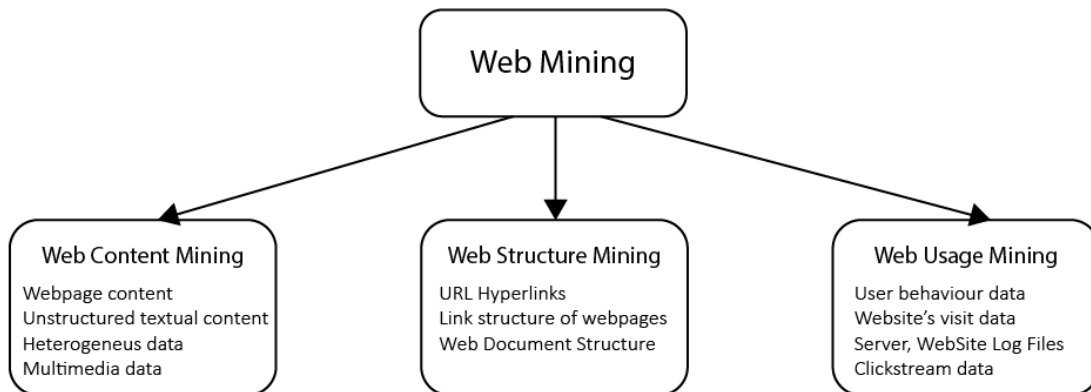


Figure 3.1: Web Mining Categories

3.2 Web Usage Mining

Web usage mining refers to the discovery of user access patterns from Web usage logs. The kind of data collected can be data stored in server access logs, referrer logs, agent logs, user profiles, bookmark data, user sessions, cookies stored by browsers, and any other data generated by the interaction between the user and the web page.

Web Usage Mining have an important part in understanding the usability of the website design, the improvement of user's relations and improving the necessity of system presentation [MA15].

One important kind of data is the data generated by users' click stream. This data after analysed can be very useful to predict user behaviour and to redirect the user to the information which he demands more easily.

The main sources to get the row log data from a website are: i) Client Log File; ii) Proxy Log File; and (iii) Web Server Log File.

3.2.1 Web Server Log File

The most significant and frequently used source for web usage mining is web server log data. This web log data is generated automatically by a web server

Table 3.1: Web Server Log File Types and Content

Log File Type	Content
Access Log	All resource access request sent by user
Agent Log	User's browser, version, OS
Error Log	Details of errors occurred while processing user access request
Referrer Log	Contains information about referrer page.

when its services user requests, which contains all information about visitor's activity [KBoo]. The common server log file types are access log, agent log, error log and referrer log [SK]. Table-1 summarizes each.

3.2.2 Web Usage Mining Tasks

The process of web usage mining can be divided in three independent tasks [CMS99, Shr12]: **Preprocessing**: do the conversion of the raw data into the data abstraction (users, sessions, episodes, clickstreams, and pageviews) necessary for further applying the data mining algorithm; **Pattern Discovery** which converges the algorithms and techniques from data mining, machine learning, statistics and pattern recognition; **Pattern Analysis**: validation and interpretation of the mined patterns.

3.2.3 Preprocessing

The preprocessing of Web usage mining is usually complex. Purpose of preprocessing is to offer structural, reliable and integrated data source to pattern discovery. It consists of four steps: (i) Data cleaning, (ii) Transaction Identification; (iii) Data Integration; (iv) Transformation [CMS99].

Data Cleaning

Data cleaning is the first step performed in the preprocessing of Web usage mining. The entries which are irrelevant in data analyzing and mining are removed. In data cleaning process, firstly, entries that have status of "error" or "failure" should be removed. Secondly, some access records generated by automatic search engine agent should be identified and removed from the access log.

Transaction Identification

After the data cleaning, the log entries must be partitioned into logical clusters using one or a series of transaction identification modules. These transaction identification modules can be called as either a merge or a divide module. The aim of transaction identification is creating meaningful clusters of references for each user. Both types of modules take a transaction list and possibly some parameters as input, and as output a transaction list that has been operated on by the function in the module in the same format as the input [CMS99].

Data Integration

The integration of content, structure, and userdata in other phases of the Web usage mining may also be essential in providing the ability to further analyze and reason about the discovered patterns.

Transformation

Once the domain-dependent data transformation phase is completed, the resulting transaction data must be formatted to conform to the data model of the appropriate data mining task. For example, the format of the data for the association rule discovery task may be different than the format necessary for mining sequential patterns [CMS99].

3.2.4 Pattern Discovery

After data preparation phase, the pattern discovery method should be applied. This phase consists of different techniques derived from various fields such as statistics, machine learning, data mining, pattern recognition applied to the Web domain and to the available data. The task for discovering the patterns offer some techniques as statistical analysis, association rules, sequential pattern analysis, clustering and so on. Here we will briefly describe some techniques to discover patterns from processed data [Shr12].

- Statistical Analysis such as frequency analysis, mean, median.

- Clustering of users help to discover groups of users with similar navigation patterns (provide personalized Web content).
- Classification is the technique to map a data item into one of several predefined classes.
- Association Rules discover correlations among pages accessed together by a client.
- Sequential Patterns extract frequently occurring intersession patterns such as the presence of a set of items followed by another item in time sequence.
- Dependency Modelling determines if there are any significant dependencies among the variables on the website.

3.2.5 Pattern Analysis

Pattern Analysis is the final stage of Web Usage Mining which involves the validation and interpretation of the mined pattern.

Validation

To eliminate the irrelevant rules or patterns and to extract the interesting rules or patterns from the output of the pattern discovery process.

Interpretation

The output of mining algorithms is mainly a mathematic form and not suitable for direct human interpretations.

3.3 Web Usage Mining Approaches

In [VMKR13], Varnagar et al. identify some work done on data collection and pre-processing stage of web usage mining. They state that Web log data pre-processing is very important and a crucial task in the entire mining process and can be strengthened by choosing and neatly applying various heuristic techniques. Further, the author adds that exploiting the usefulness of both client and server

side log data, is useful to produce result that are more efficient and a better match to empirical observations.

3.4 Web Analytics

One of the most common form of analysis of web usage data is the statistical analysis. The data is aggregated by predetermined metrics such as sessions, visits, domains, page views.

Studying the behaviour of the website users, can lead to an optimization of usage and of the user experience through the website. To measure the different events of the website, web analytics tools use different Metrics. Metrics are different types of available user information, are used as a way of analysing Web traffic and can help improving a Website to meet its goals. These metrics can be divided in four categories: website usage, referrers, website content analysis, and quality assurance [SKS14]. The Figure 3.3 [BJo8] shows some examples of metrics related to each of these categories.

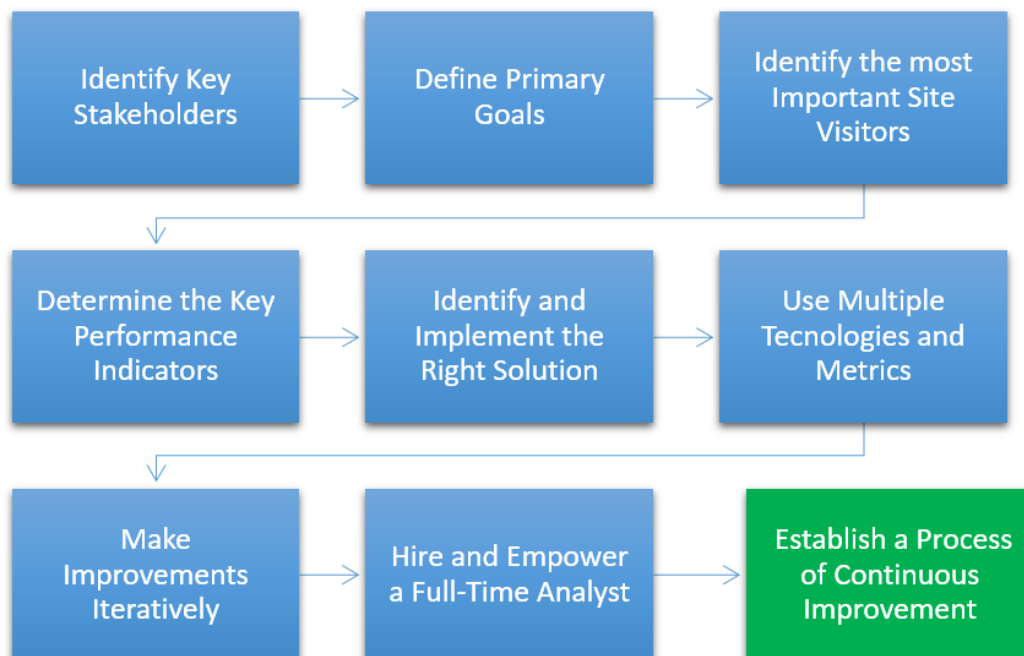


Figure 3.2: Best Key Practices: Web Analytics Process Guide [BJo8, Chr]

Web analytics deals with several methods for measurement, data collection and data analysis. These are the main feature of commercial solutions available

for web usage analysis.

The web analytics process involves several steps to be implemented. First, must be defined the goals, i.e. the objectives for each metric defined. Then it is necessary to gather and collect de usage data from several sources that are considered relevant. With this collected data, an analysis for each metric is done and finally changes are implemented.

Site Usage	Referrers	Site Content Analysis	Quality Assurance
<ul style="list-style-type: none"> • Numbers of visitors and sessions • How many people repeatedly visit the site • Geographic information • Search Engine Activity 	<ul style="list-style-type: none"> • Which websites are sending visitors to your site • The search terms people used to find your site • How many people place bookmarks to the site 	<ul style="list-style-type: none"> • Top entry pages • Most popular pages • Top pages for single page view sessions • Top exit pages • Top paths through the site • Effectiveness of key content 	<ul style="list-style-type: none"> • Broken pages or server errors • Visitor response to errors

Figure 3.3: Metrics Categories

3.4.1 Tools

There are several tools of web analytics to collect and analyse web usage data from a website. Google Analytics [goo15], Piwik [piw15], CrazyEgg [cra15] are some of the most used web analytics tools in the market. They have different capabilities and functions and also differ in the kind of data they collect.

Although there are several tools, they focus mainly on statistical analysis of website data usage [KSK12] not providing detailed reports as intended in this proposal.

By collecting various Web analytics metrics, such as number of visits and visitors and visit duration, one can develop key performance indicators (KPIs - Key Performance Indicators) – a versatile analytic model that measures diverse metrics against each other to define visitor trends. KPIs use these dynamic numbers to get an in-depth picture at visitor behavior on a site. This information allows businesses to align their Websites' goals with their business goals for the purpose of identifying areas of improvement, promoting popular parts of the site, testing new site functionality, and ultimately increasing revenue [BJo8].

The kind of data gathered is also related to the metrics and KPIs that each tool analyses and gives report. There are several metrics and KPIs available in each tool, however, there are some that are present in web analytics tools:

- Conversion Rate: KPI that measures the percentage of total visitors to a Website that perform a specific action
- Exit page: The pages visitors viewed last on the site.
- Landing Page: The pages through which visitors entered the site.
- New Visitor: A user who is accessing a Website for the first time
- New Visitor Percentage: KPI that measures the ratio of new visitors to unique visitors.
- Repeat Visitor: A user who has been to a Website before and is now returning
- Returning Visitor: KPI that measures the ratio of unique visitors to total visits
- Total Bounce Rate: KPI that measures the percentage of visitors who scan the site and then leave.
- Unique Visit: One visit to a Website (regardless of if the user has previously visited the site); an alternative to unique visitors.
- Unique Visitor: A specific user who accesses a Website.

3.5 Discussion

Information like number of visits, duration of visit, page views, click path and exit rate, can be used for different purposes as assessment of quality of web-products (i.e., websites, web videos) [SKS14] or to statistical analysis of website data usage [KSK12].

This set of data can be collected through web analytics tools and can be helpful to maintain the requirements, particularly with regard to traceability, extensions and prioritization that, ultimately, can help in the improvement and maintenance of the web systems.

Currently, web analytics tools are able to collect diverse data about the usage of a website but they only generate reports with navigation statistics, duration

of navigation on the website and other metrics that are mostly often used simplistically to see which content has more adherence by users [Ver11]. Either, web analytics has focused on analysis and reporting of business metrics of interest mainly to marketers [PRRS13].

Despite of continuous research in the field of Web Analytics, there are still some challenges which researchers need to work upon. The analysis and research done till this moment show that some of the open challenges of this field are:

- There are several web analytics tools with large volumes of data, but they do not give any recommendations to the improvement of the website based on the data collected.
- Stakeholders are often skeptical regarding a new form of automated tool support [MT13]. Therefore, high quality recommendations should be presented to answer the stakeholder's doubts.
- No focus on the improvement of the software requirements specification (SRS).
- The current web analytics tools have a number of weaknesses. For instance, they do not analyze the service based on different kinds (roles) of users; do not analyze typical navigational paths (which may be useful to define or improve workflows); do not produce reports based on the requirements and, therefore, in a language closer to the business.

Web Mining has an immense potential in itself for recommendations to users based on their preferences [Ver11]. Web usage mining is also a field with immense potential itself for recommendations for systems developers to help in the website improvement [KSK12].

The potential of the analysis of usage data is yet to be explored [ABB12]. An analysis directed to the improvement is not currently done and this data is disregarded for the development and improvement of the quality of a web application. Web Analytics tools available [SKS14] do not provide functionalities that enable a more intelligent analysis to suggest improvements to the website, such as suggest new workflows, identify and remove unused features or present more readable and legible reports.

3.6 Summary

Web usage mining has emerged as the essential tool for realizing more personalized, user-friendly and business-optimal Web services.

Moreover, Web Usage Mining may be used to support user personalization, web site design and other business making decision.

The key is to use the user-clickstream data for many mining purposes. Traditionally, Web usage mining is used by e-commerce sites to organize their sites and to increase profits. It is now also used by search engines to improve search quality and to evaluate search results, and by many other applications.

—A point of view can be a dangerous luxury when substituted for insight and understanding.

Marshall McLuhan

4

Recommender Systems

4.1	Recommender Systems	52
4.2	Collaborative filtering	52
4.3	Content-based filtering	53
4.4	Hybrid Recommender Systems	53
4.5	Recommender Systems for Software Engineering	55
4.6	Recommender Systems Approaches	56
4.7	Discussion	58
4.8	Summary	59

Recommendation (or Recommender) Systems (RS) for Software Engineering is a novel approach to support developers in decision making.

Robillard, Walker, and Zimmermann [RWZ10] define a Recommendation System for Software Engineering (RSSE) as a:

"A software application that provides information items estimated to

be valuable for a software engineering task in a given context."

RSSEs can help developers to find alternative decisions in a wide range of software engineering tasks from reusing code to writing effective bug reports.

4.1 Recommender Systems

In recommendation systems, the utility of an item is usually represented by a rating, which indicates how a particular user liked a particular item. However, in general, utility can be an arbitrary function. Depending on the application, this function can either be specified by the user, it is often done for the user-defined ratings, or is computed by the application, as it can be the case for a profit-based utility function (such as the one on ebay.com or amazon.com) [MT09].

Recommendation Systems have been widely used in e-commerce websites to provide user personalization [PB07] like product, content or service recommendations.

Recommendation Systems are commonly classified on three categories, based on how recommendations are made [BS97]:

- Content-based recommendations: Items similar to the ones preferred by the user in the past will be recommended to such user
- Collaborative filtering (or collaborative recommendations): Items that people with similar tastes and preferences liked in the past will be recommended to such user
- Hybrid approaches: Combine collaborative and content-based methods

4.2 Collaborative filtering

Is an implementation of word-of-mouth promotion where purchase decisions are taken on the basis of the opinion of relatives and friends: if users A and B rated similar items in a similar fashion in the past, Collaborative Filtering will propose new items to user A that B already rated positively [GFD⁺14, GKV14].

4.3 Content-based filtering

Perform item recommendations by predicting the utility of items for a particular user based on how "similar" the items are to those that he/she liked in the past [Liu11].

Examples:

- In a movie recommendation application, a movie may be represented by such features as specific actors, director, genre, subject matter, etc.
- The user's interest or preference is also represented by the same set of features, called the user profile.

Recommendations are made by comparing the user profile with candidate items expressed in the same set of features. The top-k best matched or most similar items are recommended to the user [BHCoo]. The simplest approach to content-based recommendation is to compute the similarity of the user profile with each item [GGF14].

4.4 Hybrid Recommender Systems

Hybrid recommender systems combine two or more recommendation techniques to gain better performance with fewer of the drawbacks of any individual one. Most commonly, collaborative filtering is combined with some other technique in an attempt to avoid the ramp-up problem [Buro7].

Two main problems have been addressed by researchers in this field, cold-start problem and stability versus plasticity problem. Cold-start problem occurs when learning based techniques like collaborative, content-based, and demographic recommendation algorithms are used [GS09, ER12].

4.4.1 Taxonomy of Hybrid Recommendation Systems

Burke [Buro2] divided hybrid recommendation systems in seven categories: (i) weighted, (ii) switching, (iii) mixed, (iv) feature combination, (v) feature augmentation, (vi) cascade, and (vii) meta-level:

- **Weighted hybrid** – This hybrid combines scores from each component using linear formula. Therefore, components must be able to produce its recommendation score which can be linearly combinable. Also, the components have to be consistent relative accuracy across the product space and to perform uniformly.
- **Switching hybrid** – The issue of this hybrid is selecting one recommender among candidates. This selection is made according to the situation it is experiencing. The criterion for the selection like confidence value or external criteria should exist and the components might have different performance with different situations.
- **Mixed hybrid** – This is a hybrid which is based on the merging and presentation of multiple ranked lists into one. Each component of this hybrid should be able to produce recommendation lists with ranks and the core algorithm of mixed hybrid merges them into a single ranked list. The issue here is how the new rank scores should be produced.
- **Feature combination hybrid** – There exist two very different recommendation components for this hybrid, contributing and actual recommender. The actual recommender works with data modified by the contributing one. The contributing one injects features of one source to the source of the other component.
- **Feature augmentation hybrid** – This is similar to the feature combination hybrids but different in that the contributor generates new features. It is more flexible and adds smaller dimension than feature combination method.
- **Cascade hybrid** – This one is a tie breaker. The secondary recommender is just a tie breaker and does refinements.
- **Meta-level hybrid** – For this one, contributing and actual recommenders exist but the former one completely replaces the data for the latter one, not just part of it.

There are several recommendation systems that use a hybrid approach by combining collaborative and content-based methods, which helps to avoid some limitations of content-based and collaborative systems [BS97, BHC00, Yun, SPUP02].

Different ways to combine collaborative and content-based methods into a hybrid recommender system can be divided as follows [AT05]:

- implementing collaborative and content-based methods separately and combining their predictions,
- incorporating some content-based characteristics into a collaborative approach,
- incorporating some collaborative characteristics into a content-based approach, and
- constructing a general unifying model that incorporates both content-based and collaborative characteristics.

4.5 Recommender Systems for Software Engineering

Currently, exist several research work in the field of recommendations systems of software engineering [MRE12]. They are mainly focused on the coding phase where the developed tools give recommendations to assist developers on several programming tasks like suggesting code reuse opportunities [YF05] or support software requirements elicitation [CHCH10].

An RSSE might need to provide or infer all the following aspects as part of the context [RWZ10]:

- the user's characteristics, such as job description, expertise level, prior work, and social network;
- the kind of task being conducted, such as adding new features, debugging, or optimizing;
- the task's specific characteristics, such as edited code, viewed code, or code dependencies; and
- the user's past actions or those of the user's peers, such as artifacts viewed and artifacts explicitly recommended.

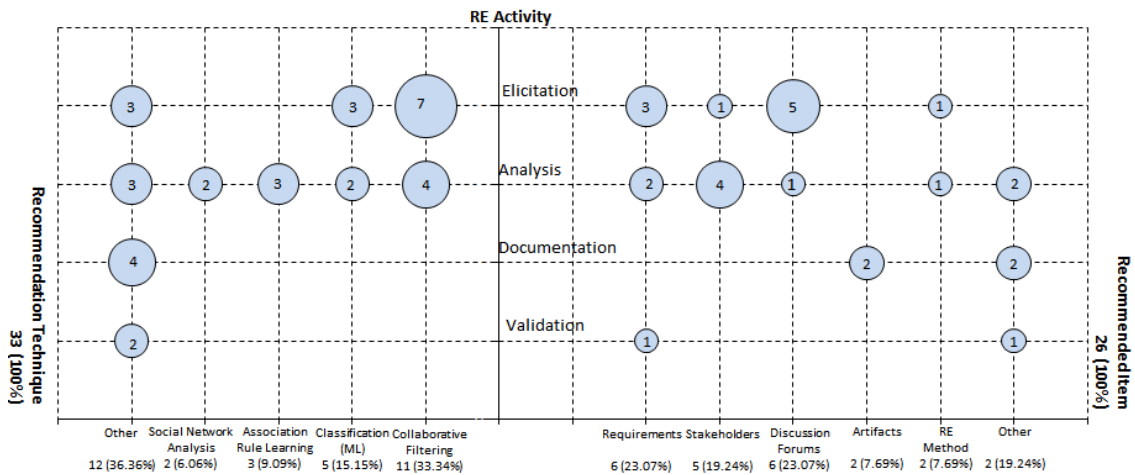


Figure 4.1: Combined mapping of the RE activity, recommendation technique, and type of items recommended [MRE12]

In Figure 4.1 it is shown a combined mapping of each activity of Requirements Engineering with the recommendation technique and the type of items recommend by each recommendations system of software engineering analysed in a systematic mapping study of RSSEs [MRE12]. Most of the RSSEs analysed address the elicitation and analysis activities of the Requirements Engineering process with very few studies related to the validation technique.

4.6 Recommender Systems Approaches

There are Recommendation Systems that have been widely used in e-commerce websites to provide user personalization [PB07] like product, content or service recommendations. Recommendation Systems provide information items estimated to be also valuable for a software engineering task in a given context [RWZ10]. Recommender Systems for Software Engineering (RSSE) is a novel approach to support developers in decision making.

RSSE can help developers to find alternative decisions in a wide range of software engineering tasks from reusing code to writing effective bug reports. The overall goal is to provide the right information, at the right time, to the right person. This would allow requirements engineers to spend their limited time on more important aspects of the project [RWZ10].

Ghezzi [GPST14] presented an approach that automates the acquisition of

user-interaction requirements through web logs and analyses them by means of probabilistic model checking to identify navigation anomalies and emerging users behaviours.

SRRS [RMZR08] proposes a Security Requirements Recommendation System that uses prior knowledge about security requirements approaches, combined with user preferences, to recommend the most appropriate approach for specific project characteristics, and in fact present a full-order ranking of all approaches. Castro-Herrera [CH10] presented a hybrid Recommendation System to identify potential users who could reply to unanswered posts in online forums. A content-based recommender technique analyses the text of unanswered posts, and compares them to previous posts. Cleland-Huang [CHM08] proposed a feasible approach that utilizes data mining and recommender systems to scale-up the fundamental processes of requirements elicitation and prioritization.

Castro-Herrera [CHCH09] presented a novel technique, that utilizes data mining techniques to analyse stakeholders' contributions to a project in order to identify potential experts. A method for automatic classification of informal requirements is presented by Ko, Youngjoong et al. [KPSC07], which is particularly useful in large-scale projects. StakeNet [LQF10] is an approach for stakeholder identification which is based on the concepts of social network analysis. Fitzgerald et al. [FLF12] developed an approach to feature requirements management based on predicting software failures (e.g., abandoned implementation of a feature) by analysing the communication threads in feature management systems.

In a survey related to software requirements specification in model-driven development, Valderas [VP11] demonstrated that few of the existing approaches are specifically defined for the specification of web application requirements. Furthermore, once requirements are specified, there is little support for allowing the systematic or automatic derivation of the conceptual model that properly satisfies the software requirements specification.

Currently there are several research works in the field of recommender systems for software engineering (RSSE) [MRE12]. They are mainly focused on the coding phase where the developed tools give recommendations to assist developers on several programming tasks like, suggesting code reuse opportunities [YF05] or support software requirements elicitation [18]. For instance, Maalej [MT09]

proposed a continuous and context-aware approach for communicating user input to engineering team.

4.7 Discussion

Research in the field of Requirements Management based on data collected of the usage of website is few [GP14], or not exactly related with the research proposed on this research work. For instance, Gao [GLX⁺11] proposed a solution where the evolution of software requirements models is based on the feedback collected.

In addition, nowadays, the Requirements Management of a website is only done to solve detected problems, satisfy or correct the failures of the software requirements specification [KMA⁺13] not focused in its improvement, and therefore in the improvement of the quality of the website.

Several studies [MT13, RWZ10] have been developed related to Recommender Systems of Software Engineering , however none of them produce recommendations to the Software Requirements Specification based on the web usage data of the website.

In other related work, Danylenko and Lowe [DL12] suggest recommendation systems based on context-aware composition to enable a system designer to postpone and automate decisions regarding efficiency non-functional requirements, such as performance. Though, this approach focus on handling efficiency non-functional requirements during the software development process and is carried out before deploy of the system.

A recommendation-based approach to requirements reuse is presented by Dumitru [DGH⁺11]. The proposed recommendation approach is content-based filtering, where a vector of keywords (derived from the description of the new software project) is matched with the keywords extracted from requirements artifacts from the repository of already completed software projects. A recent recommender system, INTELLIREQ [FRST15] is based on different recommendation approaches that support stakeholders in requirements-related activities such as definition, quality assurance, reuse, and release planning. However, on this approach, the recommender system is used just through the elicitation phase.

4.8 Summary

The available RSSE [MRE₁₂, LADR₁₂] differ from our proposal since the data collected to generate recommendations is not based on the data on the use of the website. In addition, our approach differs from all the related works described, because we have developed a recommender system (REQAnalytics) that supports the evolution of the website and allows the requirements maintenance throughout its lifecycle.

Despite the fact of several studies [RWZ₁₀] [MT₁₃] [RMWZ₁₄] have been carried related to RSSEs, none of them produce recommendations to the Software Requirements Specification based on techniques of web mining or based on web usage data.

Using the data gathered from a web analytics tool, REQAnalytics generates recommendations to the software requirements specification and to the website itself. These recommendations also allow to analyze typical paths taken by users (which may be useful to define or improve workflows) and suggest changes to the functional requirements and to the website in a language closer to the business.

Part II

Research Approach and Solution

—*We are drowning in information
but starved for knowledge.*

Rutherford D. Rogers

5

Research Problem

5.1 Research Challenges	64
5.2 Thesis Statement	65
5.3 Research Questions	65
5.4 Research Goals	66
5.5 Validation Methodology	68
5.6 Summary	68

Uncontrolled, incomplete or incorrect requirements lead to many software project failures [Sta95, KS09, LFVV11]. Programmers and software organizations are improving the methods they use to gather, analyze, document, and maintain their requirements. Nowadays, project teams document their requirements in a structured software requirements specification written in natural language. However, a Software Requirements Specification has some limitations [Wie99]:

- It is difficult to keep current

- It is hard to communicate changes to the affected team members
- It is difficult to store supplementary information about each requirement
- It is hard to define links between functional requirements and corresponding use cases, designs, code, tests, and project tasks.

Requirements management tools can diminish these limitations since these tools provide functions to store requirements and related information, import and export requirements and define links between requirements. However, requirements management tools, particularly when used with the requirements of web applications, which are permanently evolving during its lifecycle, disregard one important kind of information that is the web usage data.

Web usage data is commonly used to generate reports with the navigation statistics, duration of navigation on the website and other metrics of interest mainly to increase the number of visits of the website. In this research work, it is intended to use web usage data to support requirements maintenance in order to improve the quality of the website.

In this chapter, there are identified and described several research challenges that led to the proposed novel approach of supporting the requirements management of a website through a recommender system using the web usage data. Then the thesis statement of this research work is defined and explained. The chapter concludes presenting the methodology that was used to validate the thesis.

5.1 Research Challenges

There are still some challenges which researchers need to work upon. The analysis and research done till this moment show that some of the open challenges of this field are:

- It is extracted little knowledge of the data collected.
- There are many web analytics tools with large volumes of data, but they do not give any recommendations to the improvement of the website based on the data collected.

- Stakeholders are often skeptical regarding a new form of automated tool support [MT13]. Therefore, should be delivered high quality recommendations to answer the stakeholders's doubts.
- No focus on the improvement of the software requirements specification.
- Available tools support some kind of analysis, however, are not able to perform other kind of analysis. For instance, do not analyse typical paths taken (which may be useful to define or improve workflows); do not produce reports based on the requirements and, therefore, a language closer to the business.

In a service context in which maintenance gains importance, the analysis of the use of the service may also be useful for the maintenance of the requirements as it may, for example, suggest prioritization changes which may have an impact on new updates/changes of software system.

5.2 Thesis Statement

Considering the literature review presented in Chapter 2, 3 and 4, the study and analysis performed, and the research challenges identified in Section 5.1 (p. 64), the thesis statement is defined as:

The analysis of the use of a web evolving system can identify possible recommendations for changes in requirements that will improve the system quality and can help maintain the requirements (e.g., traceability, prioritization) with further improvements in the own system

5.3 Research Questions

Given this thesis statement, the research work is focused on the following research questions:

- **Q1** *How web usage mining can support requirements change management?*
- **Q2** *How to create traceability links between the functional requirements and the website application?*

- **Q3** *How to analyze web usage data in order to suggest new navigation paths related with the functional requirements?*
- **Q4** *How to generate recommendations to the software requirements specification?*
- **Q5** *What type of recommendations can be suggested to functional requirements?*
- **Q6** *How to provide more readable reports in a language closer to the business?*

5.4 Research Goals

Based on the problem and motivation identified during exploratory research and the raised research questions, there were defined a primary goal and two secondary goals in order to satisfy these research questions.

5.4.1 Primary Goal

The primary goal of this research is to *define a novel approach to Requirements Management through a Recommender System, REQAnalytics that collects information on the usage of a web application, relates that information back to the requirements, and generates reports with recommendations and change suggestions that can increase the quality of that service.*

The main requirements of this approach are:

- **Integrated Recommender System** to provide the ability to visualize in a single application the information related to the functional requirements correlated with its mapping information and the recommendations generated by the system.
- **Maintain the requirements specification updated** by analyzing the use of the website to identify possible gaps between the website implementation and its specification.
- **Analyze multiple websites.** Using a web analytics tool that can collect the web usage data of different websites, REQAnalytics allows to analyze multiple websites.

- **Improve the development of the system**, supporting the evolution of the system, by proposing changes and improvements to the requirements specification.

5.4.2 Secondary Goals

There are two secondary goals derived from the primary goal of this research work.

Secondary Goal - Mapping Tool

On this secondary goal is intended to *develop a mapping tool that maps the functional requirements of the software requirements specification of a given website with the web pages and elements of the website.*

This goal should be achieved considering the following requirements:

- **Reduce the manual work** required to maintain the traceability relationship information;
- **Maintain traceability relationships** between the requirements and the implementation artifacts;
- **Identify the implemented functional requirements** presented in the software requirements specification of a website and therefore trace functional requirement coverage.

Secondary Goal - Recommendation Reports

This secondary goal aims to *generate readable recommendation reports based on web usage data of a website in a language more closer to the business than the reports provided by existing web analytics tools.*

The main requirements of this approach are:

- Generate Recommendations to the Functional Requirements:
 - **Priority** of change of a requirement

- **Create** new requirement
- **Delete** existing requirement
- **Split** existing Requirement
- **New requirement dependency**
- **Remove requirement dependency**

5.5 Validation Methodology

This research work was validated through two empirical case studies to compare it with other approaches, and to prove the approach feasibility. Further, the results obtained during the research work have been presented and discussed in international conferences and published in international journals, after being approved in its reviewing processes.

5.6 Summary

Requirements Management tools are widely used to document and to maintain the software requirements specification updated. Nevertheless, when the software product is a service provided continually over time, like websites, it is very common to emerge new requirements or changes to existing requirements. Hence, requirements management tools disregard web usage data that can be useful to understand the relevance of the requirements in the requirements specification namely its priority and if there exist requirements whose functionalities are no longer accessed by website users among others.

There are many web analytics tools with large volumes of data, but they do not give any recommendations to the improvement of the website based on the data collected. Despite of continuous research in the field of Web Analytics, there are still some challenges which researchers need to work upon. Available tools support some kind of analysis, however, are not able to perform other kind of analysis.

In a service context in which maintenance gains importance, the analysis of the use of the service may also be useful for the maintenance of the requirements as

it may, for example, suggest recommendations to the requirements specification like change the priority of the requirements, create new requirements for functionalities that are not documented, remove requirements of functionalities that are not used, which may have an impact on new updates/changes of software system.

The next chapter presents the proposed approach.

*—The alchemists in their search for gold
discovered many other things of greater
value.*

Arthur Schopenhauer

6

REQAnalytics

6.1 Overview	73
6.2 Architectural Design	75
6.3 Web Analytics Tool Selection	76
6.4 Mapping Tool	79
6.5 Collect Web Usage Data	83
6.6 Analysis of the Data Collected	83
6.7 Generation of Recommendation Report	84
6.8 Components of REQAnalytics	84
6.9 Summary	101

The use of websites generates large amounts of information that may be used for different purposes like assessment of quality of web-products [PRRS13] or to statistical analysis of website data usage [KSK12].

The information about the usage of websites may help software requirements

maintenance which can be a contribution to the overall quality of the service provided. Existing approaches and tools do not take advantage of this data, which could improve the process of Requirements Management of a website during its lifecycle and therefore help to improve the quality of the website itself.

This chapter focuses on the primary goal of this research work, which is to **improve the requirements maintenance based on the usage of a website using a web analytics tool and generate recommendations reports that may help the requirements maintenance and increase the quality of the software requirements specification of the website.**

This is achieved by providing software engineers and software analysts with REQAnalytics, a novel approach through a web based Recommender System that supports the task of Requirements Management. In addition, it is also presented a novel approach to create traceability links with functional requirements of a software requirements specification and the implementation artifacts of a website through a high-level web based mapping tool.

The mapping tool focuses on the secondary goal of this research work which is to **map the functional requirements of the software requirements specification of a given website with the features available in the scope of the web pages that allows to identify traceability links between the functional requirements and the developed features of the website.**

The chapter also presents how the recommendations to the requirements provided by REQAnalytics are generated. These recommendations are focused on the functional requirements of the software requirements of the website. Hence, this focus on other secondary goal of this research work which is to **generate reports of web usage data written in a language easier to read and understand to the business than the reports provided by existing web analytics tools.**

This chapter starts by presenting an overall insight on the approach, contextualizing web analytics data and how it can be used with a recommender system for generating recommendations to requirements. Then, it is presented the REQAnalytics platform that supports this approach and methodology, with an overview of its main features and describing each functionality.

6.1 Overview

REQAnalytics is a recommender system that using the web usage data of a website, and the information of the mapping of the functional requirements with the web pages and elements, suggest recommendations to the software requirements specification.

Despite having some features commonly available in a Requirements Management tool, it cannot be categorized as so, due to the fact that its goal is to complement and support the task of requirements maintenance through several recommendations to the requirements. The main features of REQAnalytics include:

- **Integration with a Web Analytics Tool.** Integrate the data provided by a Web Analytics tool in REQAnalytics and correlate this data with the functional requirements. The REQAnalytics reads the web usage data from the web analytics tool and analyze them in order to suggest recommendations to the SRS.
- **Requirements Import.** Reads the functional requirements from the software requirements specification in a XML file with the structure as described in Section 6.8.1 (p. 85) and imports them to the REQAnalytics system.
- **Mapping Tool.** This tool is integrated within REQAnalytics and relates functional requirements with the web pages and elements of the website (see Section 6.4 (p. 79)).
- **Functional Requirements List Report.** This report lists the requirements with the mapping information, i.e., for each requirement it shows the web pages and elements that are related to such requirement.
- **Details of the Requirement Report.** Shows all the details related to the requirement in REQAnalytics (e.g. Requirement dependencies; Mappings with web pages and elements; DOM element number of clicks; Most visited page comparison).
- **Dashboard Visualization Report.** A report that shows the following information: Requirements that were already mapped; Number of mapping

established; Charts indicating the percentage of the requirements of the priority ranking; Number of recommendations generated by REQAnalytics.

- **Requirements Dependencies Directed Chart.** A directed chart that shows the dependencies (detected by REQAnalytics) of each requirement with previous and subsequent requirement. The nodes of this chart represent the functional requirements and the edges represent a dependency between two requirements. This chart allows to visualize execution order of the requirements along the web application (see Section 6.8.5 (p. 91)).
- **Most used Requirements Navigation Paths Report.** A report that shows the most frequent paths taken by users along the web application. However, instead of showing the paths as sequences of web pages visited, as it is common in other tools, it shows the functional requirements executed along those paths (see Section 6.8.6 (p. 93)).
- **Traceability Matrix Report.** Correlate the links between the functional requirements and the web pages and elements of the website (see Section 6.8.7 (p. 93)).
- **Requirements Analytics (Statistics and Main Metrics) Report.** This report lists the most accessed Requirements, and other metrics like Entry Features, Exit Features and Requirements Bounce Rate.
- **Report of Recommendations to the Software Requirements Specification based on web usage data.** This report shows the main recommendations to the Software Requirements Specification. These recommendations are mainly focused in the requirements and include:
 - Create New Requirement
 - Requirements Prioritization Change.
 - Delete existing Requirement.
 - Split a Requirement in two or more Requirements.
 - New Requirement Dependency.
 - Delete Requirement Dependency.

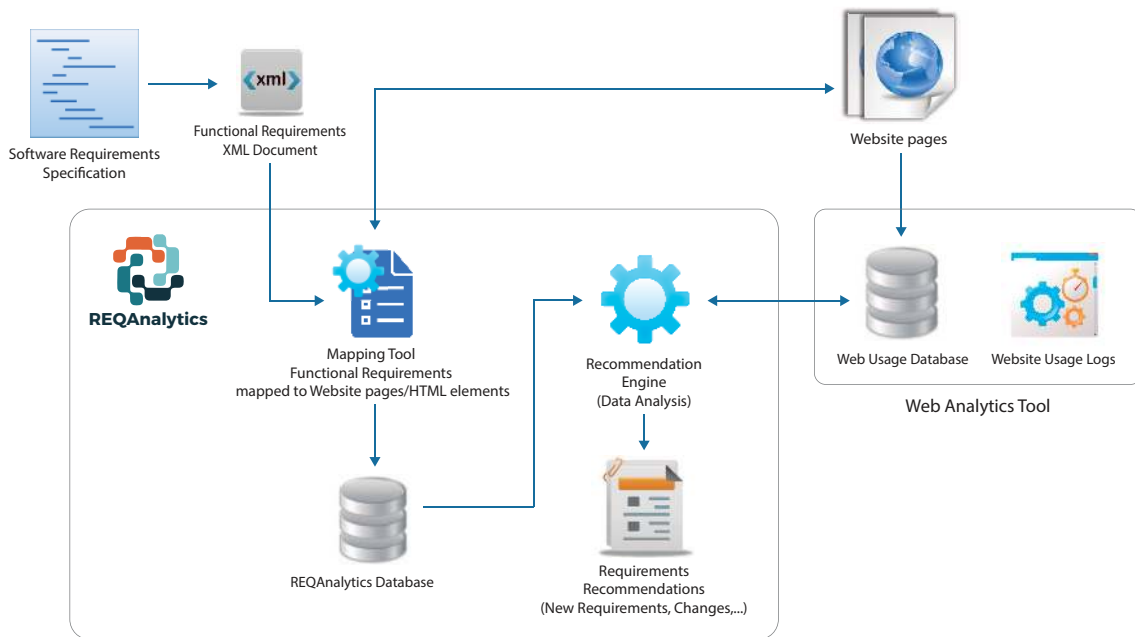


Figure 6.1: An overview of the Architectural Design of REQAnalytics recommender system

6.2 Architectural Design

This system is designed to be used totally through a web browser. Figure 6.1 shows an architecture model of the developed recommender system.

Developed in PHP and using a MySQL database as support, REQAnalytics analyses the web usage and navigation of a specific website to generate recommendations to improve the quality of the software requirements specification.

To generate these recommendations, we developed a web based mapping tool included in REQAnalytics that allows to map the functional requirements of the website with the web pages and elements of the features and functionalities of the website. To collect the web data usage from the website, REQAnalytics uses a web analytics tool, OWA (Open Web Analytics) that allows to gather this kind of data and save it to a database.

The reasons why Open Web Analytics was selected as the web analytics tool to support the task of collecting the data of web usage of the websites are detailed in Section 6.3 (p. 76).

The REQAnalytics system is divided in four different main phases:

- Requirements mapping - map the functional requirements with the web pages and elements of the website. (see Section 6.4 (p. 79))

- Collect Web usage data - use of the web analytics tool OWA for collecting web usage data (pages viewed, clicked web elements, click paths, session duration, entry pages, exit pages). (see Section 6.5 (p. 83))
- Analysis of the data collected - the data provided by OWA is analysed and intersected with the mapping information defined on the first phase. (see Section 6.6 (p. 83))
- Generation of recommendation report - it is generated a high level recommendations report with possible improvements of the requirements specification and of the website itself (see Section 6.7 (p. 84))

6.3 Web Analytics Tool Selection

The selection of the web analytics tool to use with REQAnalytics followed a process of features comparison. Based on the problem and the solution provided, the following criteria were identified and considered mandatory for the selection of web analytics tool:

- **Criteria 1:** Free or Open-Source software.
- **Criteria 2:** API and Database access to provide a way to request and work with web usage data outside of the tool.
- **Criteria 3:** Different Web Metrics and KPIs (Key Performance Indicators) to allow different types of correlations with the requirements and analysis.
- **Criteria 4:** Allow to work with several projects at same time for analyzing different web applications.
- **Criteria 5:** Track clicks in all DOM elements on each web page.
- **Criteria 6:** Identify the individual web stream paths for each user session - Navigation Paths.

Eight web analytics tools available on the market that could meet the identified criteria were examined more carefully. Table 6.1 summarizes the results of the comparison of features present in each web analytics tool.

Table 6.1: Web Analytics Tools Comparison

Web Analytics Tool	Crit. 1	Crit. 2	Crit. 3	Crit. 4	Crit. 5	Crit. 5
Google Analytics	Yes	Yes ¹	Yes	Yes	No	No
Open Web Analytics	Yes	Yes	Yes	Yes	Yes	Yes
Piwik	Yes	Yes	Yes	Yes	No ²	Yes
Woopra	Yes ³	Yes ⁴	Yes	Yes	No	Yes
WebTrends	No	No	Yes	Yes	No	Yes
Clicky	Yes ⁵	Yes	Yes ⁶	Yes	No	No
CrazyEgg	Yes	No	No	No	No	
Foresee	No	No	Yes	Yes	No	No

After analyzing all tools, the web analytics tool selected for gather the web usage data was Open Web Analytics (OWA), since it is the only that answers all previously required criteria and provides the features set essential for the development of this project. Figure 6.2 shows the Dashboard of the OWA tool.

One of the features considered essential was to **Track clicks in all DOM elements on each web page**, since it allows to map requirements with web pages and with other HTML elements, because OWA also collects the type of element clicked and its HTML identifier.

As shown in Table 6.1, Piwik answers almost all features except to track clicks in the DOM elements of the web page. To perform this task, Piwik requires to change the source code of all web pages of the website to insert tracker links in all HTML elements.

Tracking the clicks in all DOM elements of the web page allows to know exactly what was the clicked element and its HTML identifier. This allows to map a requirement with just a single element of a web page (e.g. input textbox, image) and not just with the URL of the web page. With this information is possible to map more than one requirement to elements inside one web page, since it is common to have more than one functionality in a single web page.

¹ No access to the database.

² Piwik allows to track DOM element clicks but requires to change the source code of website to insert tracker links in all HTML elements.

³ Limited to 30.000 user actions/month.

⁴ No access to the database.

⁵ Limited to 3.000 daily page views.

⁶ Limited to 1 website in free version.

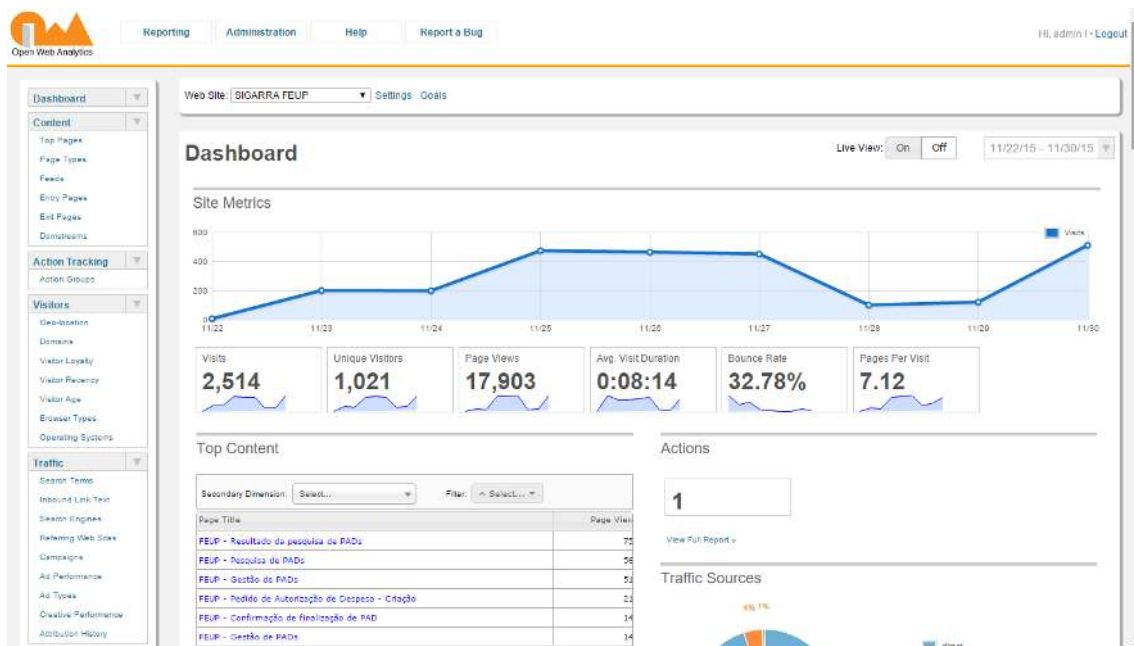


Figure 6.2: Open Web Analytics Dashboard

6.3.1 Open Web Analytics

REQAnalytics uses a MySQL database to store all the data gathered from the web usage of the websites. Since MySQL is an Open Source Relational Database Management System (RDBMS) it is possible to execute SQL queries directly in the tool. Additionally, Open Web Analytics has a REST based API that can be used to export data in JSON, XML, serialized PHP, and even basic HTML.

To collect the data of web usage, OWA uses a tracking code that needs to be inserted in each web page which is intended to analyze. This kind of procedure is similar to any web analytics tool like Google Analytics [goo15], Woopra [woo15] or Piwik [piw15].

The tracking code is a snippet of JavaScript that collects and sends data to OWA from each web page. An example of a tracking code is shown in 6.1.

Listing 6.1: Example of Javascript tracking code used by Open Web Analytics

```
<script type="text/javascript">
//var owa_baseUrl='http://www.reqanalytics.eu/';
var owa_cmds=owa_cmds || [];
owa_cmds.push(['setSiteId', 'a8e423432e9892a42124a31']);
owa_cmds.push(['trackPageView']);</pre>
</div>
```

```
owa_cmds.push(['trackClicks']);
owa_cmds.push(['trackDomStream']);

(function() {
  var _owa=document.createElement('script');
  _owa.type='text/javascript';
  _owa.async=true;
  owa_baseUrl=('https:'==document.location.protocol ?
  window.owa_baseSecUrl ||
  owa_baseUrl.replace(/http:/,'https:') z:owa_baseUrl );
  _owa.src=owa_baseUrl + 'owa.tracker-combined-min.js';
  var _owa_s=document.getElementsByTagName('script')[0];
  _owa_s.parentNode.insertBefore(_owa, _owa_s);
})();//]]>
</script>
```

6.4 Mapping Tool

This section describes the proposed approach and tool to create traceability links between software requirements of a web application and the web pages and elements of the website, similar to the one presented in [PFTV05]).

The main idea is to trace functional requirements with the implementation artifacts like web pages and elements of the website. Identifiers for each traceability element such as requirement identifiers, web page URL and HTML element identifiers are stored within the database and correlated to the functional requirements.

This mapping is a very important task in the project, since it will allow to establish correlations between the functional requirements and the implementation in order to allow REQAnalytics analyze and then generate recommendations to the software requirements specification baseline.

Figure 6.3 shows a diagram of the proposed solution. The tool runs through a bookmarklet and runs on any web browser. This allows the traceability tool to

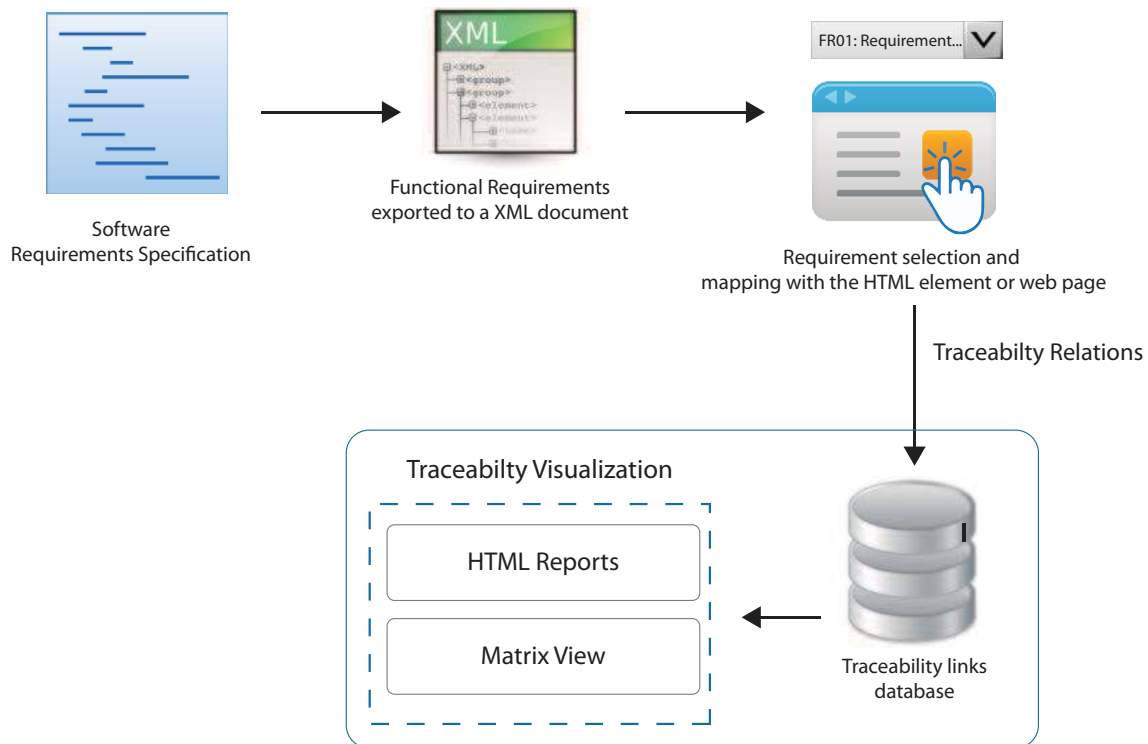


Figure 6.3: Functional requirements mapped with the web pages and elements

work with any available website on the web. To develop a tool for the proposed traceability approach we needed to divide the project in three different tasks:

- **What Information?** Identify the information to be captured and traced
- **How capture the information?** Identify how this information is captured and managed in the database
- **Development of the tool.** Create the tool itself with the different traceability views and functionalities to analyze the traced requirements with the implementation artifacts

6.4.1 What Information?

The first task toward the development of the tool was to identify the information that will be traced. Since we are analyzing the functional requirements of a web application, the traceability information necessary is the set of functional requirements, and the web pages and elements.

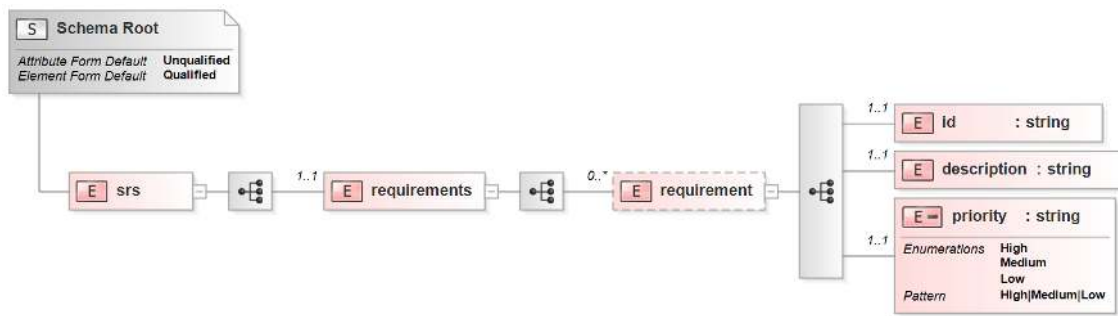


Figure 6.4: XSD Schema Diagram of the XML file necessary to import the functional requirements

6.4.2 How capture the information?

The next task was to identify how the information of the functional requirements and web pages would be captured. Firstly, to fulfill this task, the functional requirements of the website under analysis are mapped with the web pages and elements present in the website.

The mapping may be established between a single requirement and a web page, but it can also be the relation between a sequence of different web pages or elements and a requirement. With this traceability tool it is possible to map a requirement to multiple web pages and elements and an web page to multiple requirements.

6.4.3 Development of the tool

To perform this mapping, an XML document is previously generated with the functional requirements defined in the requirements specification of the website. This XML Schema was created during this research project and intends to ease the task of describing the functional requirements. This XML document must comply with the definitions of the XML Schema created for this purpose. The XML Schema Diagram is shown in Figure 6.4.

We have chosen XML on this approach due to the fact that XML is the most widely used data-interchange language among different tools and applications. Nevertheless, this could be done with another data-interchange format like JSON. Then, this XML document with the functional requirements is imported to the system to be mapped. This mapping is established manually using a high-level

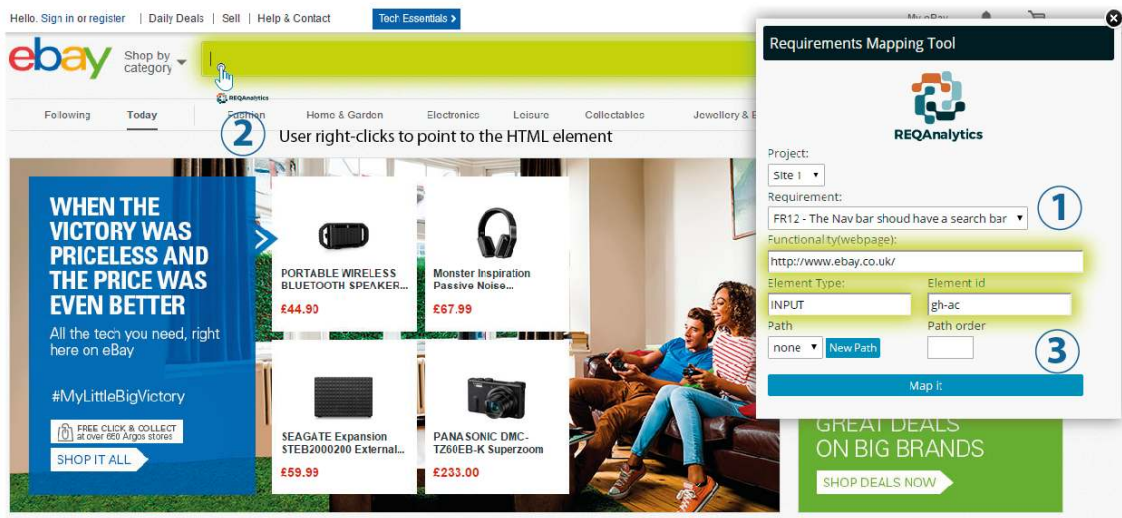


Figure 6.5: Mapping tool inside a bookmarklet

mapping web tool included in the system. As Hayes and Dekk [HD05] stated, the human analyst is required as an active participant in the traceability process.

This web tool can work with any web application or website since it was developed inside a bookmarklet that works within any web browser. A bookmarklet is a JavaScript applet that runs directly in the browser. When clicked, a bookmarklet performs a (client-side) task inside the webpage without needing to have access to the source code. To use this bookmarklet, the user that wants to perform the mapping has to access a specific link where the tool is located or just access it using a previous saved bookmark. To establish the mapping between requirements and the web elements, i.e. to create the traceability links the human interaction is needed.

A screenshot of this tool is shown in Figure 6.5.

The user must select (1) a requirement in a check box (where all the functional requirements previously imported from the XML document are), and point / click (2) on the web page and HTML element that is related with a selected requirement.

In the case of a requirement related to a sequence of web pages or HTML elements, the user has to select all the web pages and elements and map it with the requirement. Finally, by clicking on the button "Map it" (3) the tool will save the traceability link between the requirement and the web page or element. Figure 6.6 shows how the user interacts with the mapping tool.

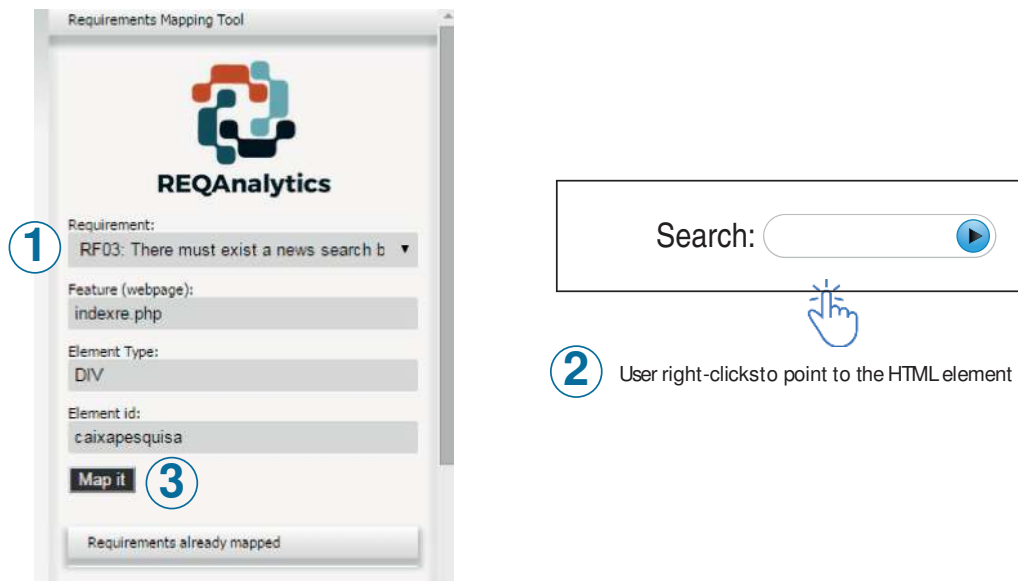


Figure 6.6: Mapping tool inside a bookmarklet to create the traceability links between requirements and the implementation

The association between the requirements and the web pages and elements is stored in a MySQL database.

6.5 Collect Web Usage Data

The purpose of this phase is to collect the web usage data available using a web analytics tool. The analytics tool used in this system as referred before is Open Web Analytics. It will collect several data like pages visited, DOM elements clicked, click paths, entry pages, exit pages and duration of session. This data will be stored in a support database for further analysis.

6.6 Analysis of the Data Collected

The REQAnalytics lists and analyses the web usage data collected by the web analytics tool, with the mapping information stored in the first phase. In this analysis, we intend to identify possible improvements of the website.

For this purpose, REQAnalytics analyse the paths taken by users while visiting the website to be able to identify, among other improvements, possible shortest paths, workflow changes, which are the most and least used functionalities (described by requirements), create new requirements not provided in the previous

requirements specification, change the priority of the requirements and detect navigation paths patterns.

6.7 Generation of Recommendation Report

After the analysis made in the previous phase, it is generated a detailed report with several recommendations for improvement of the website under review.

This report of recommendations is built in a language closer to the business (than the language used by existing web analytics tools). The recommendations presented in the report show the requirements associated with the web page and elements of the website and propose the following recommendations:

1. Show the most and least used website functionalities, mapped with the respective requirement
2. Creation of new requirements
3. Eliminate requirements whose functionality are not used
4. Change the priority of the functional requirements
5. Detect the most used navigation paths (i.e., the most used sequence of performed functionalities along the website)

6.8 Components of REQAnalytics

The REQAnalytics recommender system was developed to assist and support the task of requirements management with the objectives presented in Section 6.1 (p. 73).

The system is divided in nine main components:

- Requirements Import
- Functional Requirements List
- Details of the Requirement
- Dashboard Visualization

- Requirements Dependencies Network Chart
- Most used Navigation Paths (i.e., sequences of performed functionalities)
- Traceability Matrix
- Requirements Analytics (Statistics And Main Metrics)
- Recommendations to the Software Requirements Specification (SRS)

Each of these components is detailed below:

6.8.1 Requirements Import

The **Requirements Import** allows to import the functional requirements of the project to REQAnalytics. Figure 6.7 shows a screenshot of this component. This requirements are then stored in the database. The XML file must comply with the XML schema defined in the Figure 6.4 (p. 81).

This is the first task that the REQAnalytics' users must perform, due to the need to subsequently perform the mapping of the requirements with the implementation.

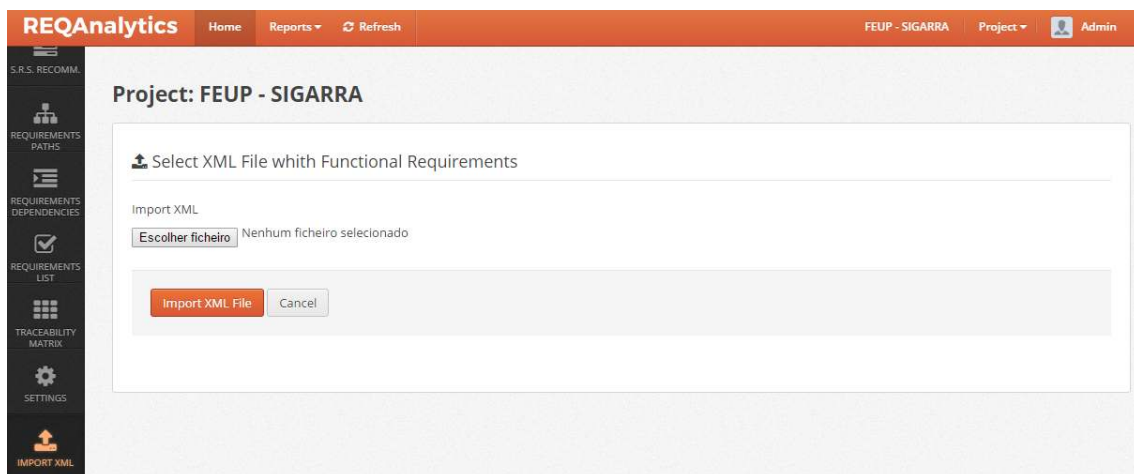


Figure 6.7: Screenshot of the component Requirements Import XML of REQAnalytics.

6.8.2 Functional Requirements List

The **Functional Requirements List** allows to list all the functional requirements that were previously imported to REQAnalytics in a table view. Figure 6.8 shows a screenshot of this component.

For each requirement, it is also displayed three attributes defined in the XML Schema defined in Section 6.4 (p. 79): Date, Status and Priority. The possible values for Status are: (i) Accepted; (ii) Not Accepted. For Priority the are three possible attributes: (i) Low; (ii) Medium; (iii) High.

ID	Title	Date	Status	Priority	Mapping
RQ01	List Active Expenditure Authorization Requests	2015-10-01	Accepted	HIGH	✓ 1 View
RQ02	Insert Document in the Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ03	Create Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ04	Search of Expenditure Authorization Requests	2015-10-01	Accepted	HIGH	✓ 1 View
RQ05	Finish Expenditure Authorization Request Confirmation	2015-10-01	Accepted	HIGH	✓ 1 View
RQ06	Classification of Expenditure Authorization Requests / Make Purchase	2015-10-01	Accepted	HIGH	✓ 1 View
RQ07	Display Statistics of Expenditure Authorization Requests	2015-10-01	Accepted	MEDIUM	✓ 1 View
RQ08	Edit the Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ09	Author changes Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ10	View Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ11	List Finished Expenditure Authorization Requests	2015-10-01	Accepted	HIGH	✓ 1 View
RQ12	Validation Data of the Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ13	Edit Item of Expenditure Authorization Request	2015-10-01	Accepted	HIGH	✓ 1 View
RQ14	List Search Results of Expenditure Authorization Requests	2015-10-01	Accepted	HIGH	✓ 1 View
RQ15	List all Expenditure Authorization Requests from a specific User	2015-10-01	Accepted	HIGH	✓ 1 View

Figure 6.8: List of the Functional Requirements previously imported to REQAnalytics

It is also viewable the mapping and the number of mappings that have been established already for each requirement. If user clicks in label with this number, it is also possible to view in a detailed tooltip what was the mapping establish, namely the HTML element id and the HTML element type (e.g. IMG, Button, A).

Figure 6.9 (p. 87) shows a cropped screenshot with an example of a tooltip with a mapping and their HTML element id and HTML element type.

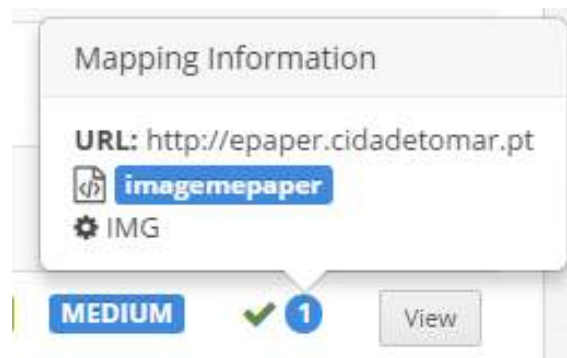


Figure 6.9: Screenshot of REQAnalytics with a Tooltip available in the Requirements List view, with a mapping and their HTML element id and HTML element type.

6.8.3 Details of the Requirement

The component **Details of the Requirement** allows the user to view several data of the requirement in REQAnalytics.

In this component the user can visualize all the information stored about the requirement stored in the database of REQAnalytics.

Figure 6.10 (p. 89) shows a screenshot of this component with a number identifying the different types of information related to the requirement. The following list enumerates these types of information:

1. **Requirement.** Identification of the requirement with their attributes: Name, Description, Status and Priority.
2. **Recommendations.** Displays the recommendations already suggested by REQAnalytics
3. **Requirement Dependencies.** Lists the requirements which depends and the dependants requirements.
4. **Mapping with web pages and elements.** Displays the mappings that have been established with this requirement using the mapping tool described in Section 6.4 (p. 79).
5. **Parameters.** In this feature of this page, the user can set if he wants to analyze this requirement or if he considers this a non-examined requirement, because despite of the web usage of this requirement, it is a mandatory requirement.

6. **Number of Visits.** Displays a chart with the number of visits on the date range of analysis.
7. **Most visited Page comparison.** Displays a chart comparing the present requirement visits with the most visited page on the date range of analysis.
8. **DOM Element Number of Clicks.** Displays a bar chart with the number of clicks in the DOM element that was mapped with requirement, on the date range of analysis.
9. **Top 5 DOM elements clicked on this page.** Displays a pie-chart with the top 5 DOM elements most clicked on this page. This chart is to show REQAnalytics' users a comparison between the DOM element that was mapped with the present requirement and the most clicked DOM elements in that web page.

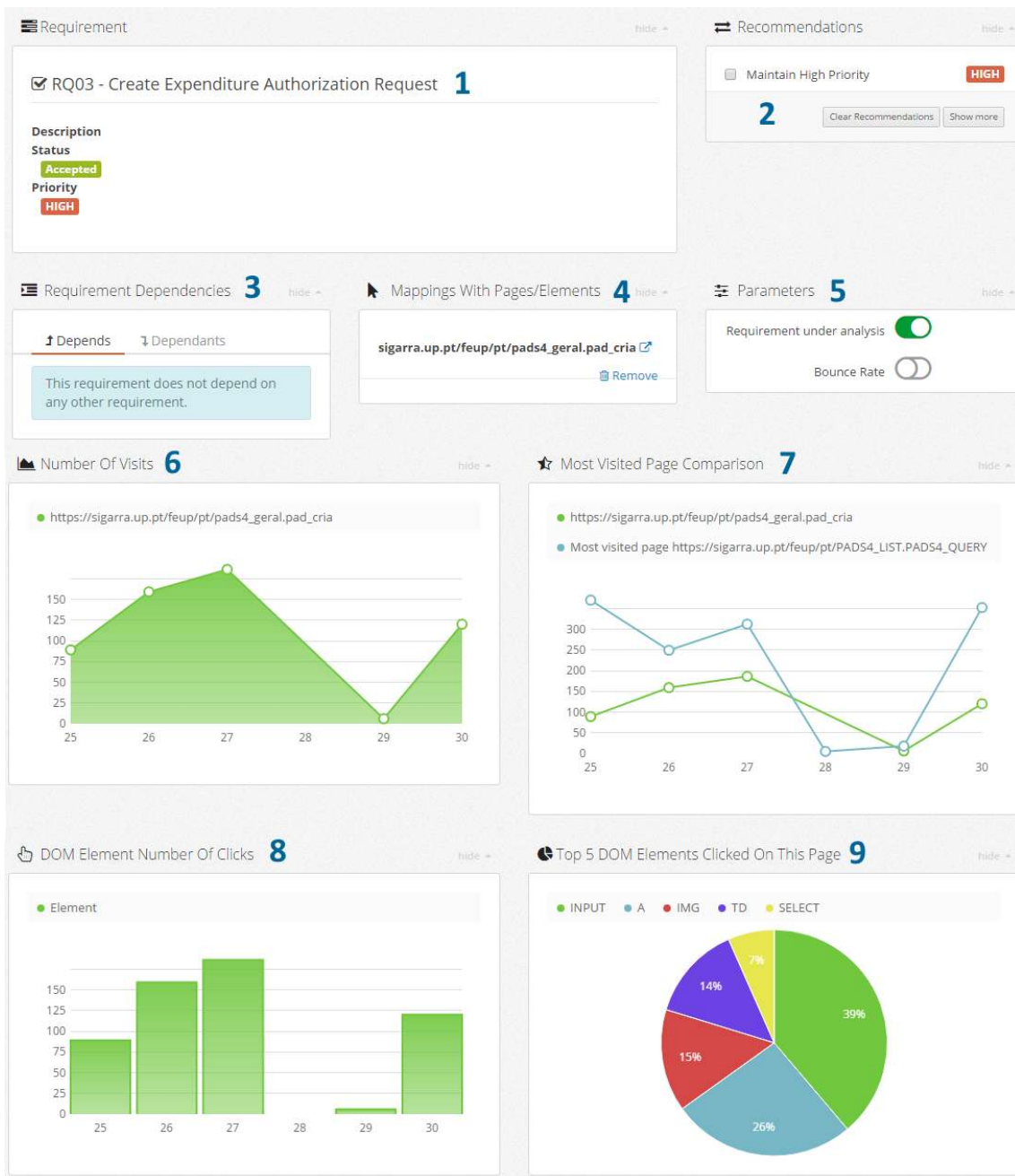


Figure 6.10: REQAnalytics component Details of Requirement.

6.8.4 Dashboard Visualization

The component **Dashboard Visualization** shows the main dashboard of REQAnalytics. This user interface allows users to check several data related to their projects. The information is organized and presented in a way that it is easy to read the most relevant information about the project. The features included in this dashboard are shown in 6.11 with the respective number in the follow list

identifying the features:

1. Number of mapping established and number of recommendations generated by REQAnalytics.
2. Bookmarklet link to include in the bookmarks of the web browser. This bookmarklet will allow to map the requirements with the web pages or elements in any website. It is also provided some instructions to help users in the task of mapping the requirements.
3. Charts indicating the percentage of the requirements of the priority ranking and percentage of mapping established and requirements mapped;
4. List of the functional requirements of the project that have been previously imported to REQAnalytics with the indication of its priority and the number of mappings that have been established with each requirement
5. Feature that allows to switch between projects that the user has privileges to access in REQAnalytics.

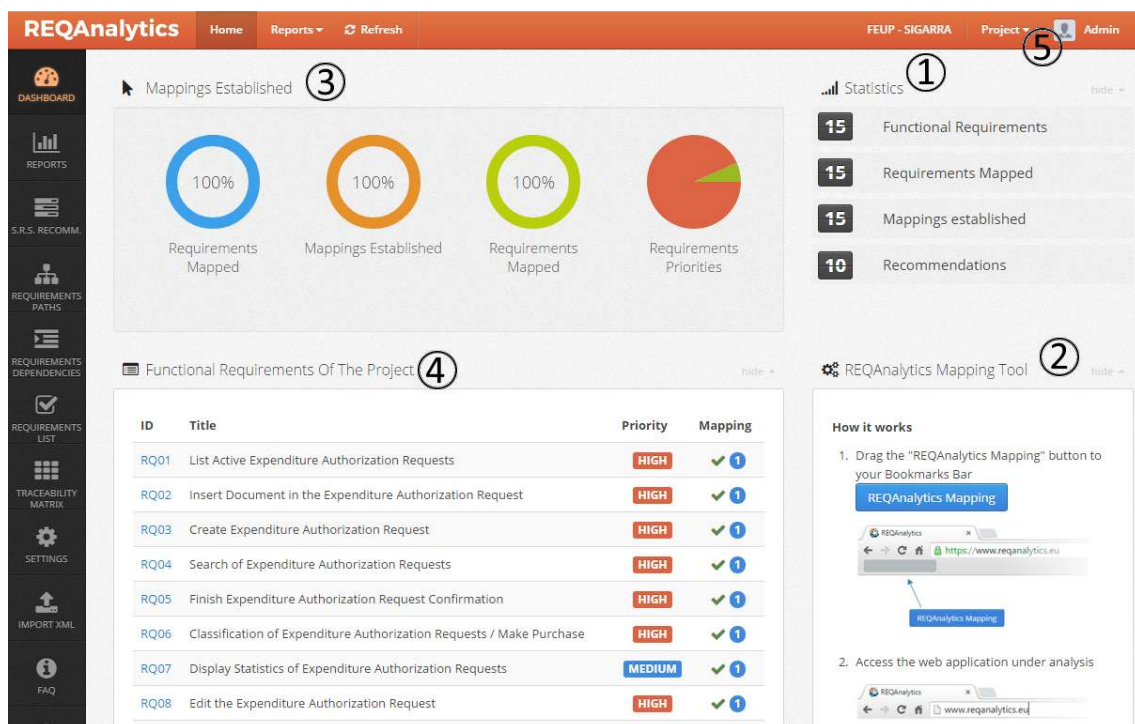


Figure 6.11: Screenshot of REQAnalytics main Dashboard

6.8.5 Requirements Dependencies Graph

The component **Requirements Dependencies Network Chart** allows to view a directed chart with all the navigations paths taken by the users of the website under analysis, from the point of view of functionalities performed (the requirements describing such functionalities).

Figure 7.15 (p. 120) shows a screenshot of the directed graph automatically generated by REQAnalytics.

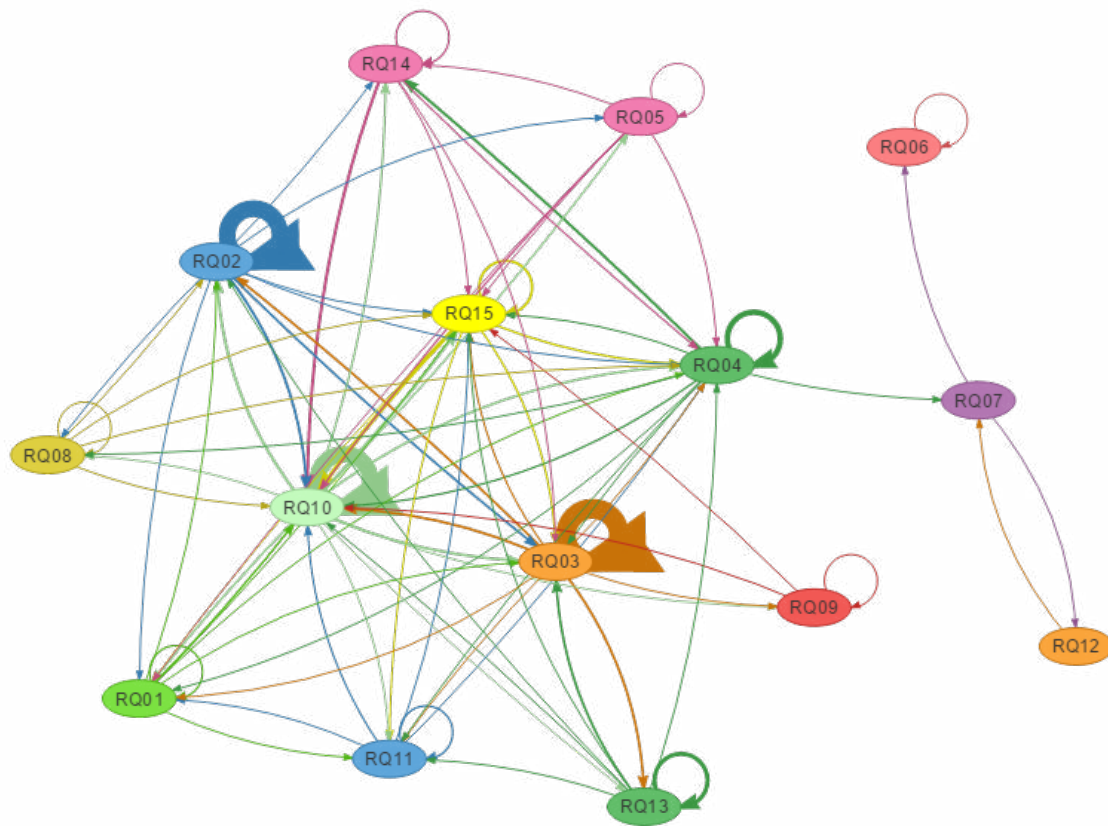


Figure 6.12: Requirements Dependencies Directed Graph

REQAnalytics automatically generates this graph based on the information of web usage supplied by Open Web Analytics correlated with the mapping information of the requirements. A Lemma 6.8.1 is shown to clear describe how this chart is generated.

Lemma 6.8.1. The mapping can be established from a requirement to a list of web elements, a set of web elements or one web element. For that we have

tree functions:

$F1 : req \rightarrow [webElement]$ \ \relates requirements with sequences of web elements

$F2 : req \rightarrow \{webElement\}$ \ \relates requirements with sets of web elements

$F3 : req \rightarrow webElement$ \ \relates requirements with web elements

We have defined a auxiliary function (F_n) to transform previous functions into one returning always a set of web elements:

$F_n : req \rightarrow \{webElement\}$

if req in $dom F1$ return $elems F1(req)$

else if req in $dom F2$ return $F2(req)$

else if req in $dom F3$ return $\{F3(req)\}$

So, now, in order to transform a sequence of logs into a sequence of requirements, we have defined a function **TransformLogsIntoSeqOfReqs** defined as:

TransformLogsIntoSeqOfReqs($log : seq\ of\ webElement$) $\implies seq\ of\ req$

if $log = []$ return $[]$

else

let $l1 \frown webElements \frown l3 = log$ in

return

TransformLogsIntoSeqOfReqs($l1$) \frown

$[r \mid r\ in\ set\ elems\ RESULT\ \&\ F_n[r] = elems\ webElement]$ \frown

TransformLogsIntoSeqOfReqs($l3$)

In the directed graph the user can for each requirement visualize its previous requirement and its post requirement that allows to identify possible new requirement dependencies. The width of each arrow is correlated with the number of clicks between the requirements.

6.8.6 Most used Requirements Navigation Paths

The component **Most used Requirements Navigation Paths** allows to view the most common navigations paths taken by the users of the website under analysis, associated with the requirements.

Figure 6.13 (p. 93) shows a screenshot of the table with the list of most used requirements navigation paths.

These requirements navigations paths were created using the same definitions described in Lemma 6.8.1.

Requirements Path	Session Paths
RQ10 → RQ10	62
RQ11 → RQ10	17
RQ10 → RQ10 → RQ10	14
RQ04 → RQ10	12
RQ10 → RQ02	12
RQ10 → RQ04	10
RQ03 → RQ03 → RQ02	10
RQ10 → RQ10 → RQ02	9

Figure 6.13: Screenshot of Most used Requirements Navigation Paths - Table view

6.8.7 Traceability Matrix

The mapping tool presented at Section 6.4 (p. 79) allows to identify traceability links between the functional requirements and the developed features of the website.

The presented methodology for requirements traceability can be applied in an evolutionary context that supports and manages functional requirements during software lifetime.

Most of existing techniques of traceability are used commonly between require-

ments and software test cases[HDO03] and these tools are unable to automatically generate and maintain traceability relationships.

With the information of the traceability links and the functional requirements stored in the database, REQAnalytics will provide several reports of traceability visualization like a traceability matrix as shown in Figure 6.14 and lists with the tracing links between the functional requirements and the implementation artifacts.

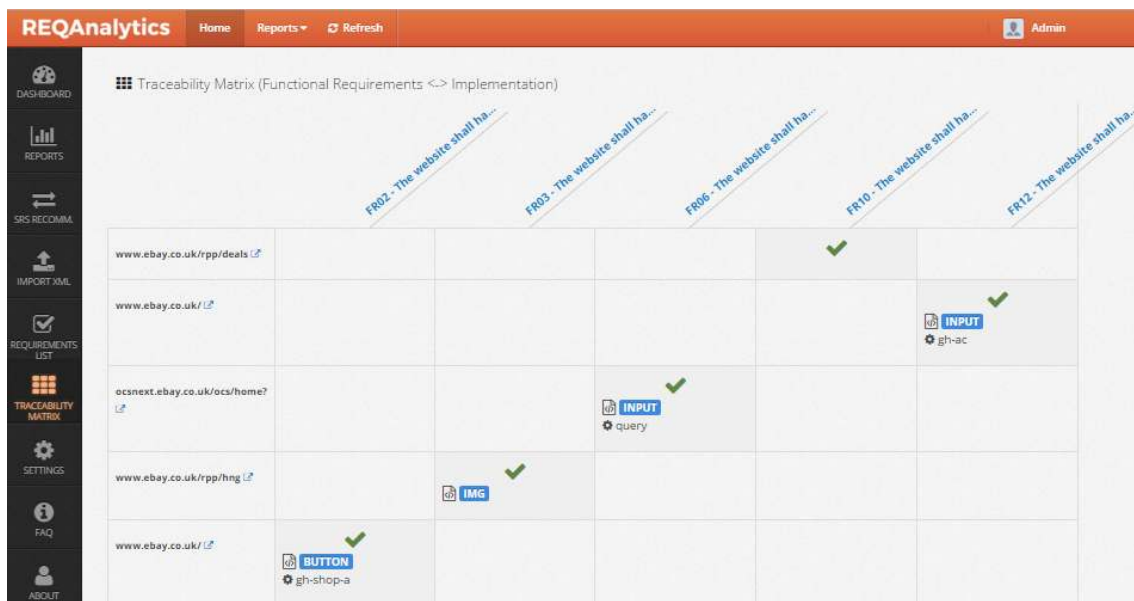


Figure 6.14: Screenshot of the Traceability Matrix

6.8.8 Requirements Analytics - Statistics and Main Metrics

The component **Requirements Analytics - Statistics and Main Metrics** displays several tables with different statistics and metrics of the functional requirements under analysis on REQAnalytics (See Figure 6.15 (p. 95)).

The metrics that are possible to view are: (i) Requirements Visits; (ii) Entry Features; (iii) Exit Features; (iv) Bounce Rate.

These metrics are similar to the metrics used by the traditional web analytics tools like Google Analytics [goo15] or Piwik [piw15]. However, on these metrics, the web pages and elements are correlated with the requirements which allow to have a novel kind report of web analytics that is the "**Requirement Analytics**".

With this novel approach in the domain of web analytics, it is possible to analyze the following also novel metrics:

- **Requirements Visits.** Displays the number of visits of the web pages and elements that were mapped with each requirement.
- **Entry Features.** Displays the number of visits as first page of the website correlated with the web pages and elements that were mapped with each requirement, i.e., the first functionality performed when starting interaction with the web application.
- **Exit Features.** Displays the number of visits as last page of user session correlated with the web pages and elements that were mapped with each requirement.
- **Requirements Bounce Rate.** Displays the percentage of visitors who enter the site and then leave ("bounce") rather than continuing on to view other pages within the same site correlated with the web pages and elements that were mapped with each requirement.

The screenshot shows a dashboard titled "Requirements Analytics (Statistics And Main Metrics)" with a "hide" button in the top right. Below the title are four tabs: "Requirement Visits" (selected), "Entry Requirements", "Exit Requirements", and "Bounce Rate". The main content is a table with the following data:

Requirement Id	Requirement Title	Clicks	Priority
RQ04	Search of Expenditure Authorization Requests	1306	HIGH
RQ14	List Search Results of Expenditure Authorization Requests	572	HIGH
RQ03	Create Expenditure Authorization Request	560	HIGH
RQ01	List Active Expenditure Authorization Requests	220	HIGH
RQ11	List Finished Expenditure Authorization Requests	134	HIGH
RQ05	Finish Expenditure Authorization Request Confirmation	131	HIGH
RQ15	List all Expenditure Authorization Requests from a specific User	115	HIGH
RQ02	Insert Document in the Expenditure Authorization Request	68	HIGH
RQ10	View Expenditure Authorization Request	50	HIGH

Figure 6.15: Screenshot of Requirements Analytics available in REQAnalytics - Statistics And Main Metrics correlated with the functional requirements

6.8.9 Recommendations to the Software Requirements Specification (SRS)

The REQAnalytics using the web usage data from Open Web Analytics and the mapping information with the correlation between the functional requirements of the website and the web pages and elements, automatically generate several recommendations that supports the task of requirements change management.

Despite, the accuracy of these recommendations, the users of the REQAnalytics need to analyze the suggested changes before implementing them into the software requirements specification of the website, because despite being sustained recommendations on real web usage data the process of recommendation is an automatic process without human intervention.

The recommender system provides the following recommendations specifically directed to the functional requirements of the website and to its maintenance and improvement:

Create New Requirement

The first type of recommendations generated by the recommender system is the creation of new requirements.

This recommendation is generated based on the web pages that have been visited for at least 10 times during the period of analysis and do not have any requirement associated.

During previous empirical experiments, we have noticed that when this occurs it is because there exists a functionality that has been probably implemented after the deploy of the website and the requirements specification was not updated accordingly.

So, the recommender system, analyzes what are the visited pages that meet these conditions and suggests the creation of new requirements as it is shown in Figure 6.16.

In addition to this analysis, the system also analyzes if there are several pages with a high similarity pattern URL (>90%). This occurs when there are pages that have the same functionality and the same name but with different values

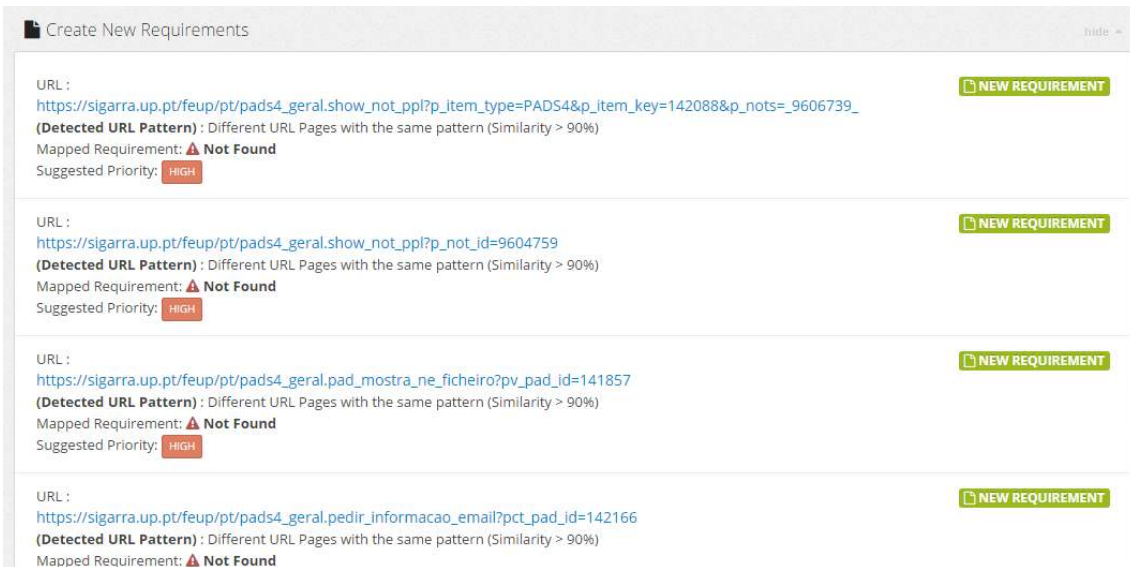


Figure 6.16: Screenshot of Recommendation to Creation of New Functional Requirements

in the URL variables. REQAnalytics aggregates these pages and suggests the creation of a single requirement for these pages.

Requirements Prioritization Change

Figure 6.17 shows the recommendations to the priority of each requirement. The possible recommendations defined for prioritization of the requirements are: Maintain, Increase or Decrease the priority of the requirement.

Rank	Requirement Id	Requirement Title	Clicks	Priority	Recommendation
1	RQ04	Search of Expenditure Authorization Requests	1306	HIGH	MAINTAIN HIGH
2	RQ14	List Search Results of Expenditure Authorization Requests	572	HIGH	MAINTAIN HIGH
3	RQ03	Create Expenditure Authorization Request	560	HIGH	MAINTAIN HIGH
6	RQ01	List Active Expenditure Authorization Requests	266	HIGH	MAINTAIN HIGH
7	RQ11	List Finished Expenditure Authorization Requests	143	HIGH	MAINTAIN HIGH
8	RQ15	List all Expenditure Authorization Requests from a specific User	141	HIGH	MAINTAIN HIGH
9	RQ05	Finish Expenditure Authorization Request Confirmation	131	HIGH	MAINTAIN HIGH
16	RQ02	Insert Document in the Expenditure Authorization Request	68	HIGH	MAINTAIN HIGH
31	RQ10	View Expenditure Authorization Request	50	HIGH	DECREASE TO MEDIUM
182	RQ13	Edit Item of Expenditure Authorization Request	18	HIGH	DECREASE TO LOW
461	RQ06	Classification of Expenditure Authorization Requests / Make Purchase	10	HIGH	DECREASE TO LOW

Figure 6.17: Screenshot of Recommendation to Change the Priority of Functional Requirements

The algorithm of the recommendation is based on the ranking of visits of each web page and elements associated with its requirement. The ranking of this approach has been set empirically based on small case studies carried out previously. First, REQAnalytics creates a ranking with the most visited requirements. After, based on this ranking, our system suggests the recommendations for the prioritization (Table 6.2).

Table 6.2: Prioritization of the Requirements

Requirement Visits Ranking	Requirement Priority
Rank 1-20	High Priority
Rank 21-50	Medium Priority
Rank >50	Low Priority

Delete existing Requirement

The second type of recommendation is to remove an existing requirement in the software requirements specification. Because of the obvious permanent consequences of this type of recommendation, it is necessary to carefully understand the need and relevance of this type of changes.

Therefore, in this type of recommendation, it is necessary to previously parametrize what are the requirements that even not having any visit during the period of the experiment, are requirements that cannot be removed. For instance, if website's Contacts page does not have any visit, it does not mean that the associated requirement should be removed. A functionality of a website or web application cannot always be measured by simplistic metrics like number of visits or its bounce rate. Hence, it is necessary to previously parametrize these settings in each functional requirement under analysis in REQAnalytics.

Then, with this parametrized precondition, the system suggests to remove the requirements which web pages do not have any visits during the period of the experimental evaluation.

Split a Requirement in two or more Requirements

This type of recommendation intends to provide users of the REQAnalytics the understanding of the web usage data of some web pages and elements that have a different behaviour when compared with other features of the website.

The REQAnalytics detects when a mapped requirement has a considerable number of visits without user leaves to other requirement of the website. Currently, REQAnalytics assumes when the number of this type of visits is above ten it is a possible case of a requirement that needs to be split.

Figure 6.18 (p. 99) shows a screenshot of REQAnalytics with an example of three requirements that should be split in two or more requirements.

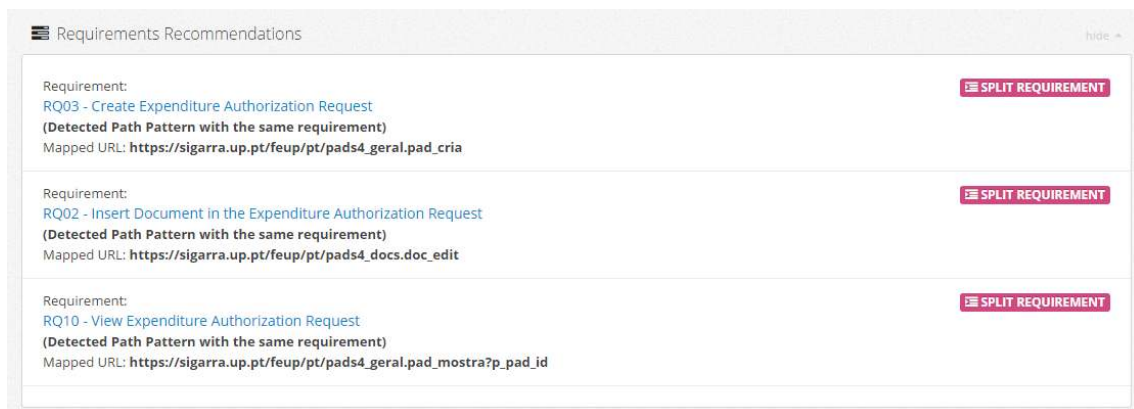


Figure 6.18: Recommendation of split a Functional Requirement in two or more New Requirements

It is assumed by the system that when this occurs it is due to three types of events that occur in the website or web application under analysis:

- **Functionalities not previously mapped.** There is more than one functionality related with the web pages and elements present which were not mapped with any requirement. However, these functionalities share the same web page of the present requirement.
- **Functionalities not initially documented in the Software Requirements Specification.** There are some functionalities that were implemented in the website that were not included in the initial specification baseline and therefore the user of REQAnalytics could not have mapped them before to include in the analysis.
- **New functionalities not documented in the Software Requirements Specification.** When there are new functionalities that were not inserted in the requirements specification.

This is an important type of analysis and recommendation provided by the REQAnalytics system since it helps to maintain the software requirements specification updated. Documenting software requirements reduce the project risk by reducing uncertainty in implementation of the software.

In addition, the documented software requirements ensures that requirements are addressed during software design and testing.

New Requirement Dependency

With the data used by REQAnalytics to generate the directed graph automatically (6.8.5), it is possible to generate more recommendations related to requirements dependencies.

With the graph created, the system has all the navigation paths taken by users of the website. Using this information, for each functional requirement, REQAnalytics knows what is the immediately preceding and succeeding requirement, i.e., the functionalities that are performed before and after a specific functionality.

Then, based in this data, it is detected what are the possible dependencies for each requirement. If a requirement A has always a previous requirement B before it means that the requirement A depends on requirement B.

Figure 6.19 shows a screenshot of an example of a recommendation to create a new requirement dependency that was detected by REQAnalytics analysing all the navigation paths.

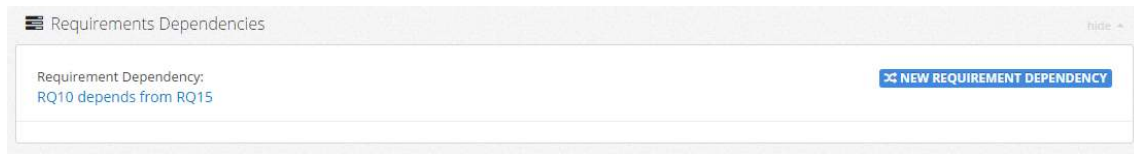


Figure 6.19: Recommendation of Creation of a New Dependency for a Functional Requirement

Delete Requirement Dependency

In an analysis similar to the analysis performed in **New Requirement Dependency**, in this type of recommendation REQAnalytics recommends to delete a specific requirement dependency.

For each requirement, it is analyzed if all the dependencies associated with it meet the dependencies detected by the system. Therefore, the REQAnalytics correlate these dependencies to find any dependency with no correlation. If it is detected that one dependency is not correlated, it is generated a suggestion to delete that requirement dependency.

6.9 Summary

This chapter presented REQAnalytics, a recommender system that, correlating the web usage data of a website and the information of the mapping of the functional requirements of its requirements specification, suggests recommendations to help identify improvements and help to maintain the website and its requirements updated which can be a contribution to the overall quality of the service provided.

It was also presented a high-level tool to map the functional requirements of a software requirements specification of a website with its functionalities, through its web pages and elements. This tool allows to identify traceability links between the functional requirements and the developed features of the website.

It is common sense in the field of software engineering that documentation of detailed and accurate software requirements contributes to the success of the systems by establishing and communicating the expectations and settings for

all aspects of the software's features and performance. Furthermore, keeping the documented software requirements updated ensures that requirements are addressed during software design and testing.

During the System Development Life Cycle (SDLC) of any software like a website or a web application, the software requirements specification is the accepted methodology when it is necessary to establish a consistent method for documenting software requirements for technology projects.

Recommender systems for software engineering can be used in a meaningful way to help the requirements maintenance during the lifecycle of a website during which requirements are constantly changing and evolving. Overall this can therefore help to improve the quality of the website.

With this kind of approach, it is possible to collect information about the usage of a website through a web analytics tool and generate recommendations reports that may help the requirements maintenance and increase the quality of the software requirements specification of the website.

The kind of analysis performed by REQAnalytics is not performed by existing web analytics tools that only generate reports with navigation statistics. In addition, the recommendations generated by REQAnalytics are presented in a language closer to the business.

Part III

Validation and Conclusions

—The greatest value of a picture is when it forces us to notice what we never expected to see.

John Tukey

7

Validation

7.1 Case Study 1: Website of Regional Newspaper	107
7.2 Case Study 2: SIGARRA: Faculty of Engineering of University of Porto Website	113
7.3 Discussion	120
7.4 Summary	123

To validate the approach proposed in this research work, including the methodology along with processes defined for the design and the development of the approach, two case studies were conducted. The first case study provides an experimental evaluation of a website. It is a website of a regional newspaper, *cidadetomar.pt*, and is described in Section 7.1 (p. 107). The second case study was carried out through an experimental evaluation of a module of SIGARRA, the web information system of Faculty of Engineering of University of Porto.

The first case study was chosen to allow to carry an experimental evaluation

on a website that is public and the type of users is very heterogeneous. This case study will allow to understand if the software requirements specification complies with the usage of a traditional news website.

In the second case study, since the module under analysis is just available in an intranet, the experiment will allow to analyse in a more controlled web environment with trained users to the core functions of the web application, how the recommendations to the requirements specification can be correct and assertive.

Both cases studies were developed following Robert Yin's guidelines [Yin09] which have nearly universal acceptance on qualitative case study methods. Yin's six-stage case study process is shown on Figure 7.1 (p. 106).

Qualitative case study has been gaining acceptance as a valid and valuable research method in a large number of diverse scientific domains [EG07, BCL11, Yin09].

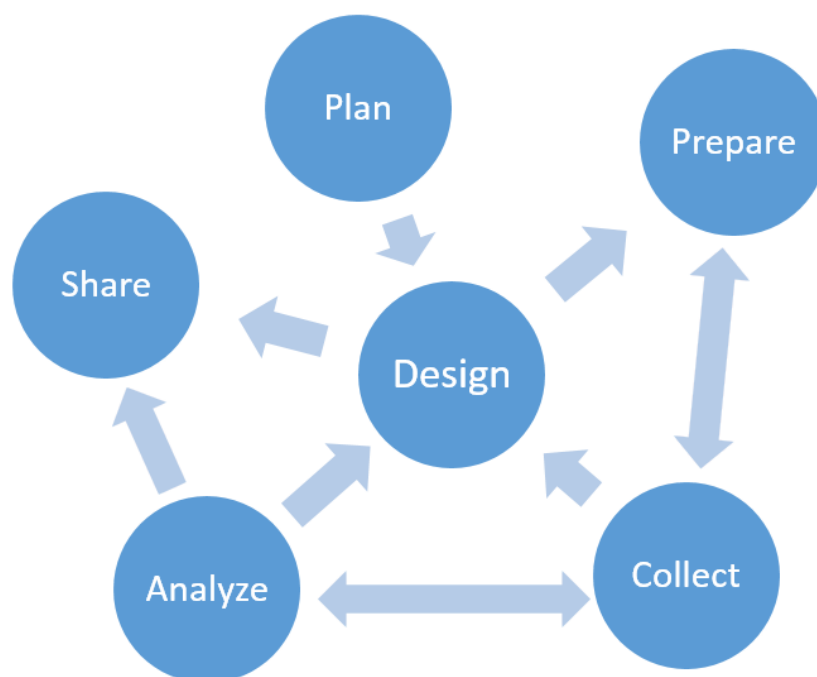


Figure 7.1: The Case Study Process from Robert Yin's guidelines [Yin09]

This method comprises all aspects of a case study research method. Therefore, when conducting a case study, there are six major activities to be followed: (1) Plan – consists in identifying the research questions and idealize the case study;

(2) Design – define data to be collected as well as procedures to maintain the case study quality (external validity and internal validity) and identify the criteria to interpret the findings; (3) Prepare – identifying teams to be part of the case study and define procedures for data collection; (4) Collect – case study execution with data gathering; (5) Analyze – consists in examining, categorizing, tabulating data to draw empirically based conclusions; and (6) Share – report the case study demonstrating its findings and results.

The objective of the first case study is to assess the REQAnalytics approach, through a news website, in order to evaluate the ability to suggest recommendations to the software requirements specification using the web usage data collected through a web analytics tool. It is also intended to evaluate what type of recommendations can REQAnalytics provide, and what are the most appropriate metrics to use in order to support the task of requirements management and maintenance.

The second case study provides a deeper analysis by evaluating the functional requirements navigation paths and the requirements dependencies so that REQAnalytics can suggest other type of recommendations to the software requirements specification.

7.1 Case Study 1: Website of Regional Newspaper

This section presents the results of the application of REQAnalytics in a news website: Cidade Tomar, a regional newspaper of Portugal.

In this case study it is intended to assess the real ability of the system to suggest recommendations to the software requirements specification and to the navigation paths. In particular, this case study is designed to answer the research questions of this research work that were defined in Section 5.3 (p. 65).

7.1.1 Goals

The main objectives defined for this case of study are:

1. Analyse the data collected from the usage of the website and produce

Table 7.1: Functional requirements analysed in the Case Study 1

Id	Description	Priority
RF01	The website must have a menu with all the categories of the news	High
RF02	The first web page must have a section with the latest news	Medium
RF03	There must exist a news search box in all web pages	Low
RF05	The website shall have a section called "Desporto"	High
RF06	The website shall have a section called "Cultura"	Medium
RF07	The website shall have a section called "Freguesias"	Medium
RF08	The website shall have a section with the Latest print edition of the newspaper	High
RF09	The website shall have a Contacts page	Low
RF10	The website must have a Home button redirecting to the first page	Low
RF11	The website should have a Disclaimer Page	Low
RF12	The website shall have a detail of each news story	High
RF13	The website shall have a section called "Opiniao"	High

recommendations to the software requirements specification and to the website itself.

2. Identify common navigation paths of sessions of the website and propose changes to the navigation paths.
3. Suggest recommendations to assign the priority to requirements under analysis.
4. Suggest recommendations to create new requirements.
5. Suggest recommendations to delete existing requirements.

7.1.2 Setup

As described in Section 6.2 (p. 75), the first phase of the REQAnalytics is to identify the functional requirements of the website requirements specification document that will be used to the analysis of the recommender system engine. Table 7.1 (p. 108) describes the functional requirements that were identified and used for analysis in this case study:

7.1.3 Procedure

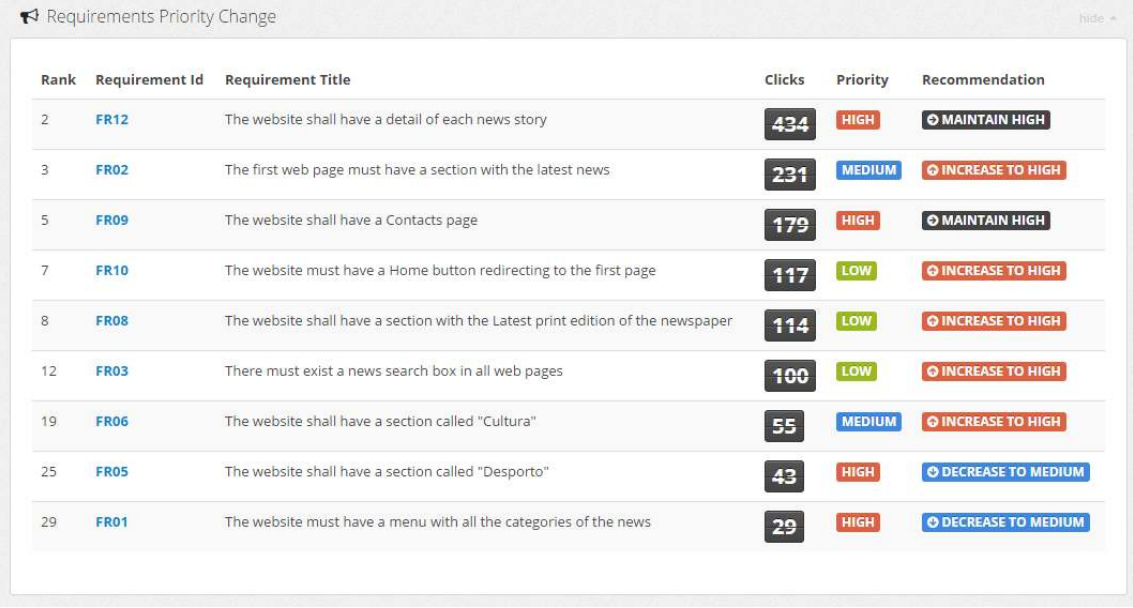
The functional requirements described in Table Table 7.1 (p. 108) were firstly exported to an XML document in the format accepted by REQAnalytics. Then the mapping between each requirement and the web page and element that implements it. This mapping was made using the web based mapping tool include in the REQAnalytics system.

The web usage data for this case study was collected using OWA. The time period used in this study was 2 weeks (14 days) before the recommender system analysis was carried out.

7.1.4 Results

With the data collected by the web analytics tool, REQAnalytics linked the web usage data with the mapping information of the requirements in order to analyze the gathered information. After this data has been analysed, REQAnalytics generated a set of recommendations to the software requirements specification:

Requirements Priority Change



Rank	Requirement Id	Requirement Title	Clicks	Priority	Recommendation
2	FR12	The website shall have a detail of each news story	434	HIGH	MAINTAIN HIGH
3	FR02	The first web page must have a section with the latest news	231	MEDIUM	INCREASE TO HIGH
5	FR09	The website shall have a Contacts page	179	HIGH	MAINTAIN HIGH
7	FR10	The website must have a Home button redirecting to the first page	117	LOW	INCREASE TO HIGH
8	FR08	The website shall have a section with the Latest print edition of the newspaper	114	LOW	INCREASE TO HIGH
12	FR03	There must exist a news search box in all web pages	100	LOW	INCREASE TO HIGH
19	FR06	The website shall have a section called "Cultura"	55	MEDIUM	INCREASE TO HIGH
25	FR05	The website shall have a section called "Desporto"	43	HIGH	DECREASE TO MEDIUM
29	FR01	The website must have a menu with all the categories of the news	29	HIGH	DECREASE TO MEDIUM

Figure 7.2: Screenshot of REQAnalytics with the Requirements Priority Recommendations in Case Study 1

The first recommendation generated by REQAnalytics was to the priority of the functional requirements shown in Figure 7.2.

Here are some of the observations/recommendations generated by REQAnalytics based on the analysed data that may help the software engineer to improve the software requirements specification of the website and the website itself.

- Recommendations to create new requirements (Figure 7.4).
- Recommendations to delete existing requirements (Figure 7.3).

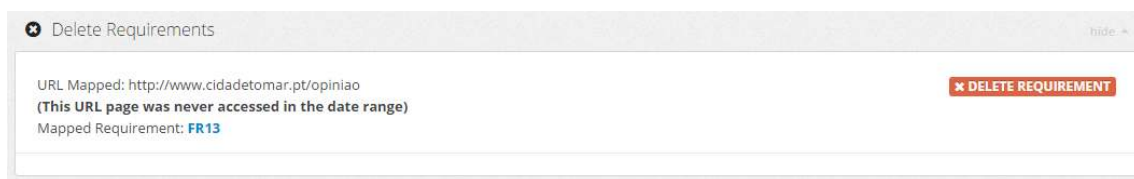


Figure 7.3: Recommendation to Delete an existing Functional Requirement based on its Web Usage

- List of the requirements analysed with the respective mapping with the web page and elements of the website. This mapping allows to identify if all requirements are implemented in the website
- Recommendations to the priority of each requirement analysed (Increase, Maintain, Lower) (Figure 7.2)
- Most used pages/functionalities of the website (Figure 7.2)
- More clicked page - `tema.php?id=1`, (4198 clicks in the period) matches the requirement with the id RF10: The portal should have a Home button that redirects to the first page. Recommendation: Increase priority of RF10 requirement to High (previous value: Low) (Figure 7.2)
- Most used navigation path of the website (Figure 7.5)

With this achieved results/recommendations, changes were proposed to the requirements specification. The changes proposed to the functional requirements of the website are shown in Figure 7.2. It was detected that requirement with the id RF10 is associated with the page `tema.php?id=1`, that was the most clicked url in the period of time analysed. Therefore, REQAnalytics recommends

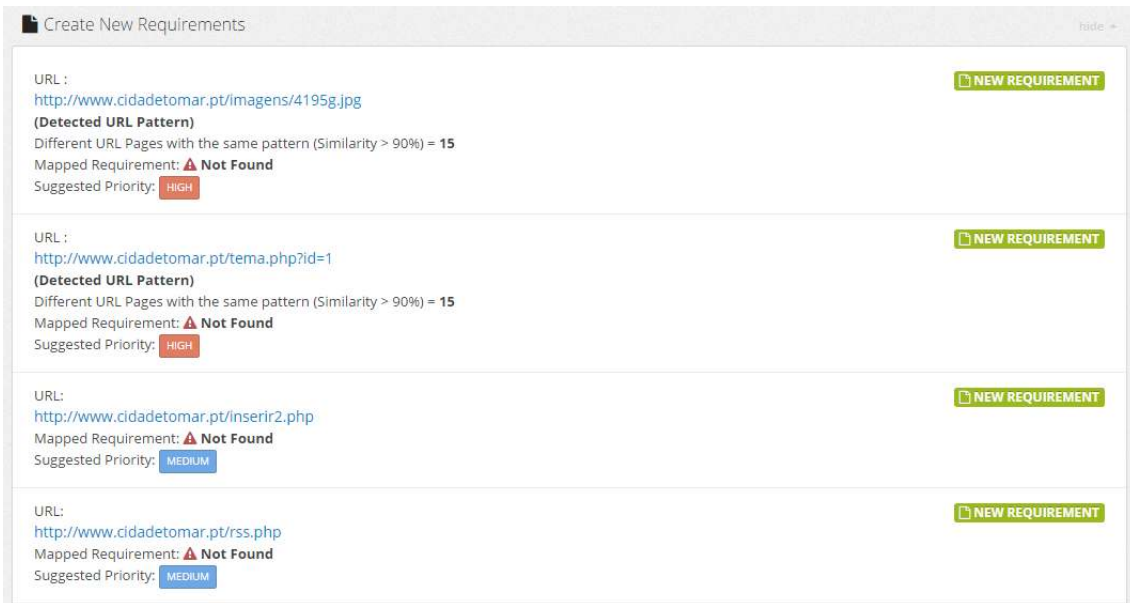


Figure 7.4: Case Study 1: Recommendation to Creation of New Functional Requirements

to increase the priority of the requirement RF₁₀ to high. These recommendations allow to support the requirements maintenance and bring the software requirements specification closer to the user needs and preferences.

Navigation Path Change

REQAnalytics also detected the most used navigation path shown in Figure 7.5 and proposed to redesign the navigation path detected.

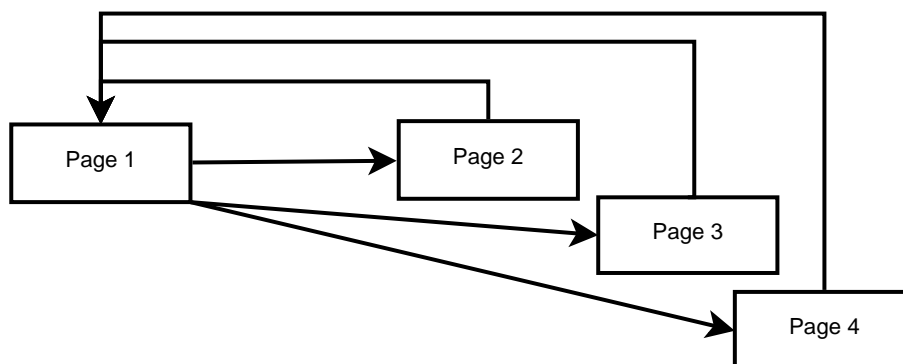


Figure 7.5: Most used navigation path.

Regarding the most used navigation path detected, it was noticed that after users clicked Page 2, Page 3 or Page 4, that were pages of news, user always need to go back to the homepage to read another news.

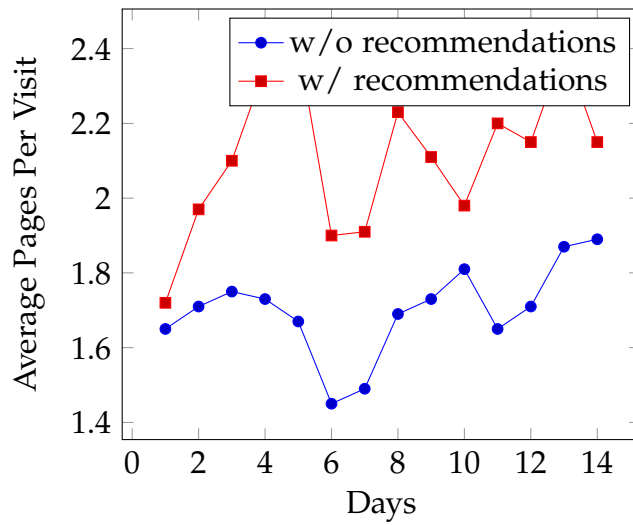


Figure 7.7: Pages Per Visit with and without recommendations applied using the same period of time, 14 days

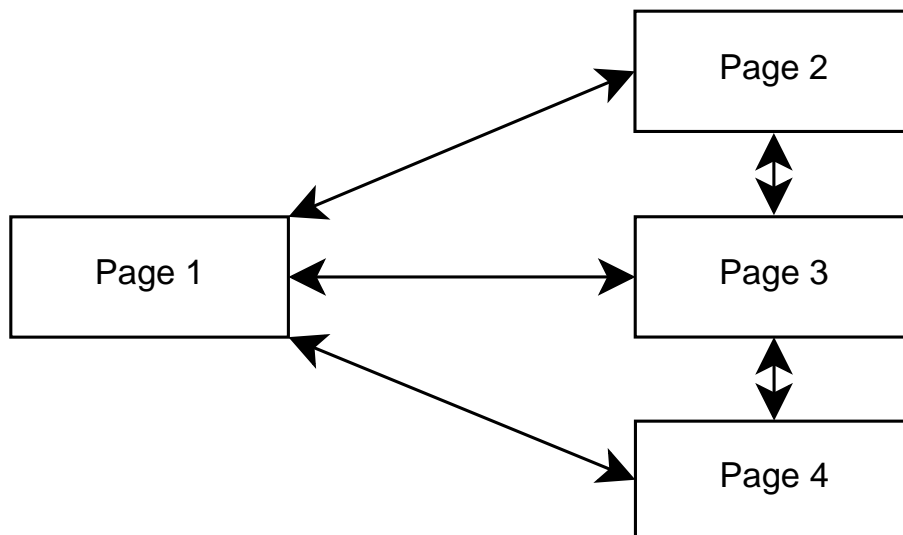


Figure 7.6: Navigation path redesigned.

Therefore, this could be a factor for the user to leave the website before it is desired, due to the excessive number of clicks the user has to perform to read the news of the website. So it was proposed to create a "Related News" section next to the news content with the navigation path presented in Figure 7.6.

After the changes made to the website, we analysed what were the improvements to the web usage of the website using the same time period of 2 weeks. The results show that before the changes the average number of pages per visit was 1.7 and after the change of the navigation path, the average number of pages per visit was 2.11 pages using the same time period of 2 weeks. Figure 7.7 shows

the results obtained for the period before the analysis carried out by REQAnalytics and the results obtained after the changes have been implemented. This corresponds to an increase of almost 25% in number of pages per visit, which for a newspaper website is a substantial value as it may correspond to an increase of their revenue from advertising.

7.2 Case Study 2: SIGARRA: Faculty of Engineering of University of Porto Website

This section presents the results of the application of REQAnalytics in the web information system SIGARRA: Faculty of Engineering of University of Porto Website. Due to the complexity of SIGARRA, it is analyzed only one Module called "Expenditure Authorization Requests Management".

In this case study is intended to assess new type of recommendations to the software requirements specification and to analyze and generate recommendations to requirements dependencies.

7.2.1 Goals

The main objectives defined for this case of study are:

- Identify the most used navigation paths of the website.
- Automatic generate a Traceability Matrix between functional requirements and implementation (Web pages and elements).
- Study what type of recommendations to the software requirements specification can be made based on the website data usage.
- Identify the most used functional requirements, the top entry features and the top exit features.
- Generate Recommendations to the Functional Requirements:
 - Change the priority of Requirements
 - Split existing Requirement

Table 7.2: Functional requirements analysed in the Case Study 2

Id	Description	Priority
RQ01	List Active Expenditure Authorization Requests	High
RQ02	Insert Document in the Expenditure Authorization Request	High
RQ03	Create Expenditure Authorization Request	High
RQ04	Search of Expenditure Authorization Requests	High
RQ05	Finish Expenditure Authorization Request Confirmation	High
RQ06	Classification of Expenditure Authorization Requests / Make Purchase	High
RQ07	Display Statistics of Expenditure Authorization Requests	Medium
RQ08	Edit the Expenditure Authorization Request	High
RQ09	Author changes Expenditure Authorization Request	High
RQ10	View Expenditure Authorization Request	High
RQ11	List Finished Expenditure Authorization Requests	High
RQ12	Validation Data of the Expenditure Authorization Request	High
RQ13	Edit Item of Expenditure Authorization Request	High
RQ14	List Search Results of Expenditure Authorization Requests	High
RQ15	List all Expenditure Authorization Requests from a specific User	High

- Create new requirement dependency
- Remove requirement dependency

7.2.2 Setup

As described in Section 6.2 (p. 75) and like in the Case Study 2 in Section 7.1 (p. 107), the first phase of this solution is to define the functional requirements of the website that will be used by the analysis performed by the recommender system engine. Table 7.2 (p. 114) describes the functional requirements that were used for analysis in this case study:

7.2.3 Procedure

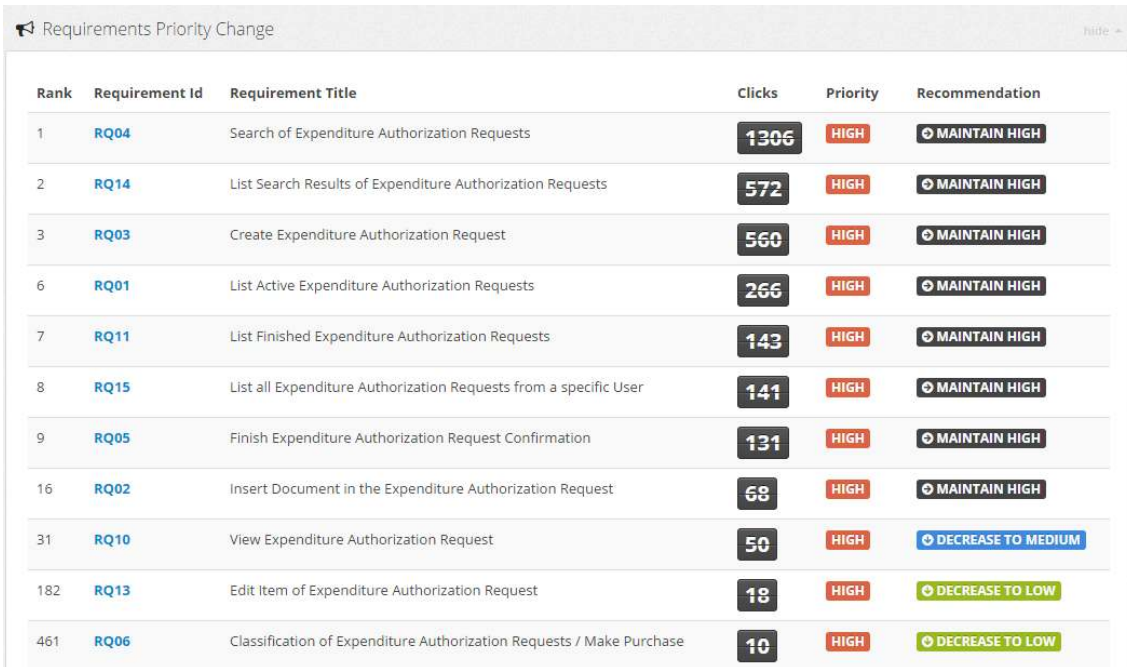
The functional requirements described in Table Table 7.1 (p. 108) were firstly exported to an XML document in the format accepted by REQAnalytics. Then

the mapping between each requirement and the web pages and elements that implements it was established. This mapping was made using the web based mapping tool include in our system and described in Section 6.4 (p. 79).

The web usage data for this case study was collected using a web analytics tool. The web analytics tool that was used to support REQAnalytics was Open Web Analytics (OWA). The time period used in this study was 2 weeks (14 days) before the recommender system analysis was carried out.

7.2.4 Results

After this data has been analysed, REQAnalytics generated a set of recommendations to software requirements specification shown in Figure 7.8.



Rank	Requirement Id	Requirement Title	Clicks	Priority	Recommendation
1	RQ04	Search of Expenditure Authorization Requests	1306	HIGH	MAINTAIN HIGH
2	RQ14	List Search Results of Expenditure Authorization Requests	572	HIGH	MAINTAIN HIGH
3	RQ03	Create Expenditure Authorization Request	560	HIGH	MAINTAIN HIGH
6	RQ01	List Active Expenditure Authorization Requests	266	HIGH	MAINTAIN HIGH
7	RQ11	List Finished Expenditure Authorization Requests	143	HIGH	MAINTAIN HIGH
8	RQ15	List all Expenditure Authorization Requests from a specific User	141	HIGH	MAINTAIN HIGH
9	RQ05	Finish Expenditure Authorization Request Confirmation	131	HIGH	MAINTAIN HIGH
16	RQ02	Insert Document in the Expenditure Authorization Request	68	HIGH	MAINTAIN HIGH
31	RQ10	View Expenditure Authorization Request	50	HIGH	DECREASE TO MEDIUM
182	RQ13	Edit Item of Expenditure Authorization Request	18	HIGH	DECREASE TO LOW
461	RQ06	Classification of Expenditure Authorization Requests / Make Purchase	10	HIGH	DECREASE TO LOW

Figure 7.8: Screenshot of REQAnalytics with the Requirements Priority Recommendations in Case Study 2

Here are some other observations/recommendations generated by REQAnalytics based on the analysed data that may help the software engineer to improve the software requirements specification of the website and the website itself.

Most Used Navigation Paths

Othe type of result achieved in this case study was the most used navigation paths related with respective functional requirement. Figure 7.9 shows a screenshot with a table with the obtained navigation paths.

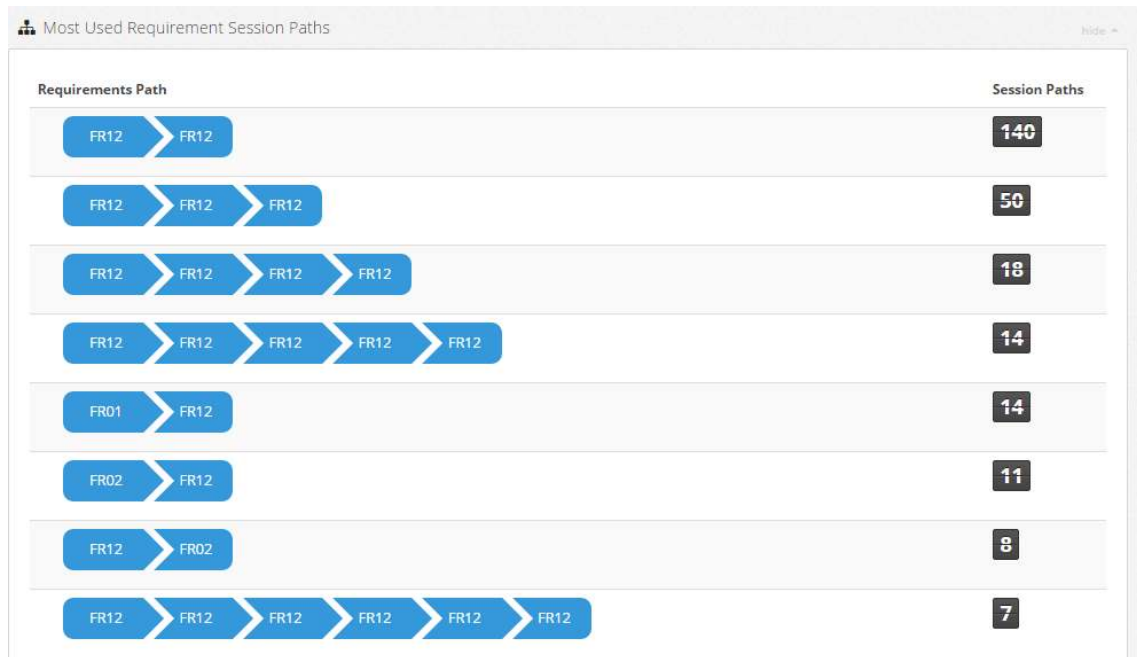


Figure 7.9: Most Used Navigation Paths related with Requirements in Case Study 2

Traceability Matrix

With the information of the traceability links and the functional requirements stored in the database, REQAnalytics was able to automatically generate a traceability matrix between functional requirements and Web pages and elements (Figure 7.10).

	RQ01 - List Active Exp...	RQ02 - Insert Document...	RQ03 - Create Expendit...	RQ04 - Search of Expen...	RQ05 - Finish Expendit...	RQ06 - Classification ...	RQ07 - Display Statist...	RQ08 - Edit the Expend...	RQ09 - Author changes ...	RQ10 - View Expenditur...	RQ11 - List Finished E...	RQ12 - Validation Data...	RQ13 - Edit Item of...	RQ14 - I...
sigarra.up.pt/feup/pt/pads4_estis.inicial						✓								
sigarra.up.pt/feup/pt/PADS4_extra.pad_auto							✓							
sigarra.up.pt/feup/pt/pads4_geral.pad_cria		✓												
sigarra.up.pt/feup/pt/pads4_geral.pad_mostp_pad_id								✓						
sigarra.up.pt/feup/pt/pads4_geral.pad_termi				✓										
sigarra.up.pt/feup/pt/PADS4_LIST.PADS4_QU			✓											
sigarra.up.pt/feup/pt/pads4_estis.class_pad_v					✓									
sigarra.up.pt/feup/pt/pads4_list.pads4_list												✓		
sigarra.up.pt/feup/pt/PADS4_list.lista_pads?p_user													✓	
sigarra.up.pt/feup/pt/pads4_geral.pad_item												✓		
sigarra.up.pt/feup/pt/pads4_docs.doc_edit	✓													
sigarra.up.pt/feup/pt/PADS4_LIST.LISTA_PADP_ESTADO=FINALIZADOS								✓						
sigarra.up.pt/feup/pt/pads4_estis.val_pad_us										✓				

Figure 7.10: Traceability Matrix between functional requirements and Web pages and elements in Case Study 2

Entry and Exit Features

In this case study it was possible to list the top entry and top exit features as defined in 6.8.8. **Top Entry Features**

Displays the top entry features of the requirements specification. The table shows the number of visits as first page of the website correlated with the web pages and elements that were mapped with each requirement.

Requirement Id	Requirement Title	Clicks	Priority
RQ01	List Active Expenditure Authorization Requests	40	HIGH
RQ15	List all Expenditure Authorization Requests from a specific User	31	HIGH
RQ04	Search of Expenditure Authorization Requests	28	HIGH
RQ03	Create Expenditure Authorization Request	11	HIGH
RQ10	View Expenditure Authorization Request	8	HIGH
RQ14	List Search Results of Expenditure Authorization Requests	7	HIGH
RQ11	List Finished Expenditure Authorization Requests	7	HIGH

Figure 7.11: Entry Features

Top Exit Features

Shows the top exit features of the requirements specification. Displays the number of visits as last page of user session correlated with the web pages and elements that were mapped with each requirement.

Requirement Id	Requirement Title	Clicks	Priority
RQ01	List Active Expenditure Authorization Requests	77	HIGH
RQ11	List Finished Expenditure Authorization Requests	26	HIGH
RQ15	List all Expenditure Authorization Requests from a specific User	24	HIGH
RQ14	List Search Results of Expenditure Authorization Requests	19	HIGH
RQ04	Search of Expenditure Authorization Requests	13	HIGH
RQ10	View Expenditure Authorization Request	9	HIGH
RQ03	Create Expenditure Authorization Request	4	HIGH

Figure 7.12: Exit Features

Split a Requirement in two or more Requirements

The REQAnalytics detects when a mapped requirement has a considerable number of visits without user leaves to other requirement of the website. Currently, REQAnalytics assumes that when the number of this type of visits is above ten it is a possible case of a requirement that needs to be split.

Figure 7.13 (p. 119) shows a recommendation of REQAnalytics of requirements that should be split in two or more requirements.

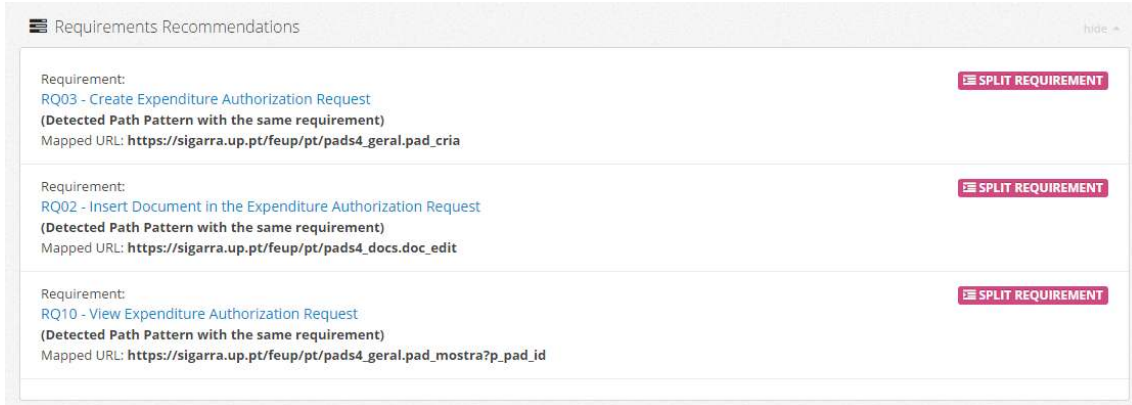


Figure 7.13: Split a Functional Requirement in two or more New Requirements

Requirement Dependencies

With the data used by REQAnalytics to generate the directed graph automatically in 6.8.5, the system automatically generates more recommendations related to requirements dependencies: (i) New Requirement Dependency; (ii) Delete Requirement Dependency.

Figure 7.14 shows a screenshot of an example of a recommendation to create a new requirement dependency that was detected by REQAnalytics analysing all the navigation paths.

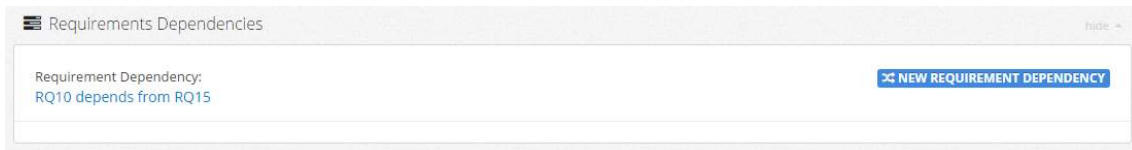


Figure 7.14: Creation of a New Dependency for a Functional Requirement

In an analysis similar to the analysis performed in **New Requirement Dependency**, in this type of recommendation REQAnalytics recommends to delete a specific requirement dependency.

Requirements Dependencies Graph

REQAnalytics automatically generates this graph with the information of web usage supplied by Open Web Analytics correlated with the mapping information

of the requirements. This graph allows to view a directed chart with all the navigations paths taken by the users of SIGARRA from the perspective of the requirements performed, i.e., functionalities described by requirements.

Figure 7.15 (p. 120) shows a screenshot of the directed graph automatically generated by REQAnalytics in Case Study 2.

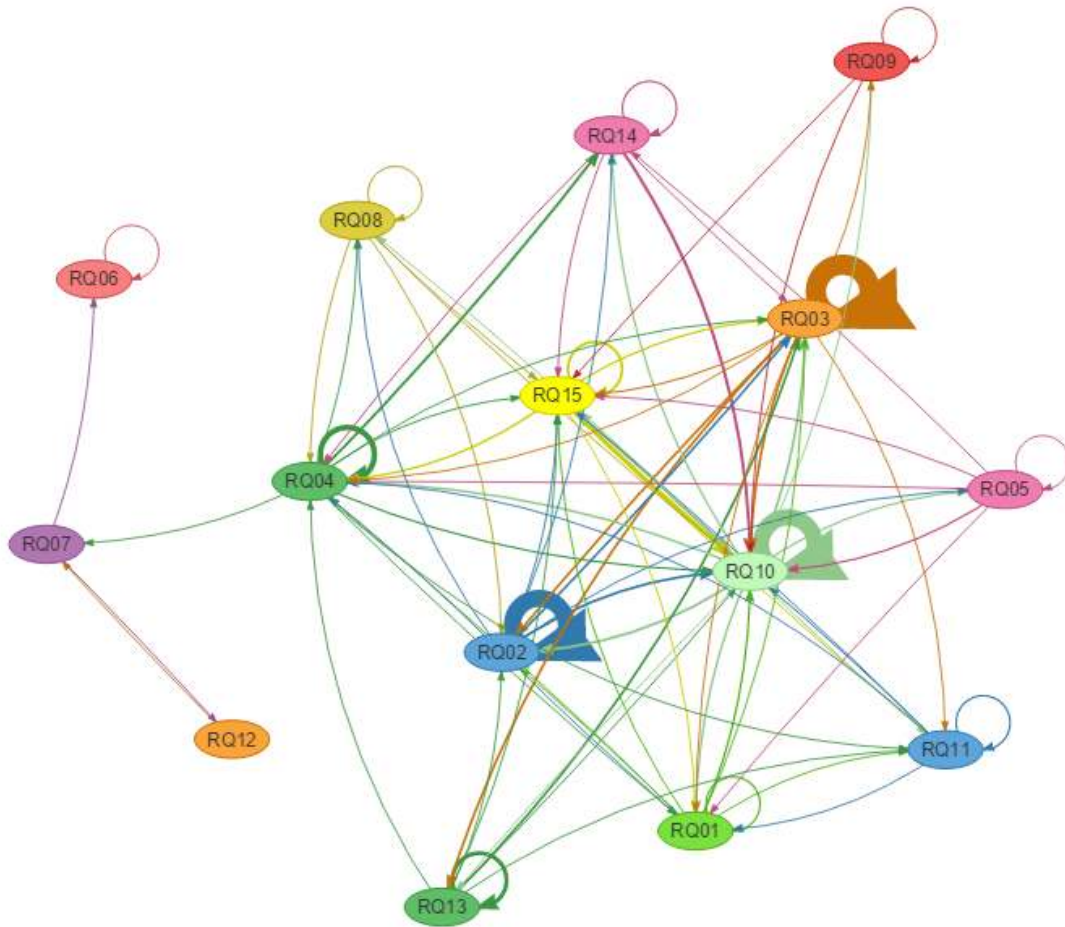


Figure 7.15: Requirements Dependencies Directed Graph of Case Study 2

7.3 Discussion

At the end of the case studies it was possible to analyze the results and findings in order to answer the research questions that were identified in Section 5.3 (p. 65) and to analyze the case study goals.

7.3.1 Case Study 1

In the first Case Study, it was possible to verify that the proposed approach is capable of analyzing the web usage of a website in order to support the requirements maintenance. Within a period of 2 weeks, the REQAnalytics system used the web usage collected by the web analytics tools to help in the task of requirements management. This results allowed to answer to the research question **Q1** - (*How web usage mining can support requirements change management?*). Analysing the web usage data as described in the proposed approach in Chapter 6 (p. 71) it was possible to understand how the behaviour of website users is important and how can it be correlated with the requirements specification.

The mapping tool used to correlate the functional requirements of the software requirements specification with the web pages and elements allowed to answer the research question **Q2** - (*How to create traceability links between the functional requirements and the website application?*). It was possible to understand the efficiency of mapping each requirement with the web pages, since the mapping tool is fully integrated in the web browser and allows to easily point and select the web pages and elements and then associate it with the functional requirements.

Regarding research question **Q2**, the REQAnalytics recommender system also provide several reports of traceability visualization like a traceability matrix and lists the tracing links between the functional requirements and the implementation artifacts (web pages and elements).

The findings of this case study also allowed to answer the research question **Q3** - (*How to analyze web usage data in order to suggest new navigation paths related with the functional requirements?*). The results show that the analysis of web usage data with the requirements allowed to redesign a navigation path of the website. The outcome showed that before these changes have been implemented, the average number of pages per visit was 1.7 and after the change of the navigation path, the average number of pages per visit was 2.11 pages using the same time period of 2 weeks.

At the end of this case study it was also possible to answer the research question **Q4** - (*How to generate recommendations to the software requirements specification?*). The results obtained show that REQAnalytics was able to suggest recommendations to change the priority of each functional requirement.

This recommendation was based on the number of accesses to the web pages and elements related with the requirements. Hence, the recommender system was able to suggest recommendations to create new requirements and suggest recommendations to delete existing requirements.

7.3.2 Case Study 2

At the end of Case Study 2, it was possible to confirm the research questions that were already answered in the first Case Study, namely **Q1**, **Q2**, **Q3** and **Q4**.

In a different website like is SIGARRA, with different purposes, since the module of this website analyzed is private and is used as a web application to support the core business of the Faculty of Engineering of University of Porto, it was possible to achieve successful results in the goals that were initially set.

The findings of this case study also allowed to answer the research question **Q4** - *How to generate recommendations to the software requirements specification?*. As in the first case study, the REQAnalytics system was able to generate several recommendations based on the web usage data gathered by OWA, like suggest recommendations to the priority change of each functional requirement and to create new requirements and suggest recommendations to delete existing requirements.

Furthermore, it was also possible to answer the research question **Q5** - *What type of recommendations can be suggested to functional requirements?*. Using the same web usage data, the results obtained show that the system generated other type of recommendations to the software requirements specification. Analyzing the web usage data and the mapping information, three other type of recommendations were identified: (i) Split existing Requirement; (ii) New requirement dependency; (iii) Remove requirement dependency.

The recommendation to split existing requirements is when REQAnalytics detects a mapped requirement with a considerable number of visits without user switching to other requirement of the website. A new requirement dependency recommendation is generated when a requirement has always a previous requirement before it is accessed. In this case it means that the requirements depend on the previous requirement. Finally, the recommendation to delete a requirement

dependency is generated when REQAnalytics cannot confirm the existence of such dependency.

At the end of this case study it was also possible to answer the final research question **Q6** - *How to provide more readable reports in a language closer to the business?*. The results obtained with this case study show that it is possible to generate several reports that allow a better understanding of the web usage data when compared to results obtained with traditional web analytics tools. The reports obtained in this case study were: (i) Functional Requirements List Report; (ii) Details of the Requirement Report; (iii) Requirements Dependencies Directed Chart; (iv) Most used Requirements Navigation Paths Report; (iv) Requirements Analytics (Statistics And Main Metrics) Report.

7.4 Summary

This chapter presented two case studies that were designed and executed to validate the REQAnalytics approach and its supporting processes.

After finishing this experimental evaluation of these two Case Studies, the recommendations given by REQAnalytics system was able to generate a set of recommendations to the functional requirements of the software requirements specification.

Changing the priority of the requirements; Creation of new requirements and Delete existing requirements.

Thus, considering this case study, the observations/recommendations given by REQAnalytics were: increase the priority of functional requirement RF10 from low to high since it was the most used functionality of the website; redefine the most used navigation path; list most used pages/funcionalities of the website and a list of the requirements analysed with the respective mapping with the web page and elements.

The recommendations generated by REQAnalytics are presented in a language closer to the business and with a kind of analysis that the existing analytics tools do not offer, which commonly only generate reports with the navigation statistics.

These results suggest that recommender systems for software engineering can be used in a meaningful way to help the requirements maintenance during the

lifecycle of a website which requirements are in constant change and evolution and therefore help to improve the quality of the website.

The next chapter discusses the research done in this work, summarizing the contributions made and points future work directions.

—We worship perfection because we can't have it; if we had it, we would reject it. Perfection is inhuman, because humanity is imperfect.

Fernando Pessoa

8

Conclusions

8.1 Summary	126
8.2 Main Contributions	127
8.3 Opportunities for Future Research Work	128
8.4 Conclusion	129

This chapter concludes this dissertation. It begins with a summary of the progresses and the main results obtained. The main contributions to academic knowledge arising from the research reported in this dissertation are after reviewed and discussed.

Taking into consideration the different research contributions, this chapter concludes pointing new opportunities for further research work.

8.1 Summary

This research work presents an approach that collects information about the usage of a website through a web analytics tool and generate recommendations reports that may help the requirements maintenance and increase the quality of the software requirements specification and the overall website.

Through a recommender system, REQAnalytics maps requirements with the functionalities of a website that implements them, and relates this information with the web usage data, in order to generate recommendations to the website under review. This mapping also allows to identify traceability links between the functional requirements and the developed features of the website.

The presented methodology for requirements traceability and software requirements management can be applied in an evolutionary context that support and manage functional requirements during software lifetime. Moreover, these techniques are used commonly between requirements and software test cases and the majority of these tools are unable to automatically generate and maintain traceability relationships.

In this research work we also demonstrated that creating traceability links between functional requirements and implementation artifacts like web pages and elements may provide a valuable help to the requirements maintenance of a website, since existing traceability methods like traceability matrices are also prone to errors and are vulnerable to changes in the system.

It also presents experimental evaluations on two case studies where some relevant results of recommendations to requirements specification of the website were achieved. These recommendations allow to change the priority of requirements and create or remove new requirements and detect shorter paths for a given website's functionality. The results obtained in the experiments suggest that recommender systems for software engineering can be used in a meaningful way to help the requirements maintenance during the lifecycle of a website during which requirements are constantly changing and evolving. Overall this can therefore help to improve the quality of the website. We also demonstrated that recommendations generated by REQAnalytics, like the detection of most used navigation paths, significantly improved the number of pages per visit in 25%.

The kind of analysis performed by REQAnalytics is not performed by existing web analytics tools that only generate reports with navigation statistics. In addition, the recommendations generated by REQAnalytics are presented in a language closer to the business.

The results show that the analysis of the use of a website provide recommendations that allow the website to meet the expectations of its customers and users. Furthermore, this approach when compared to existing web analytics and other related tools, presents more readable and understandable reports for users and stakeholders.

This approach also helps in the task of requirements management, which contributes to the quality of the web application itself. The results also indicated that the recommendations provided by REQAnalytics performed better than the random case; however additional work is needed to compare these results to those obtainable using other type of recommendations.

8.2 Main Contributions

Summarizing, the main contributions of this dissertation are:

- **An integrated Recommender System, REQAnalytics.** Provides functionalities to manage the functional requirements of a website or web application and to analyze the web usage information relating that information with the functional requirements. The recommender system is able to provide the follow contributions:
 - List of all functional requirements mapped with the web page and element;
 - Identify the most used functional requirements, the top entry requirements and the top exit requirements;
 - Detection of most used navigation paths and identification of shorter paths for a given functionality provided by the website;
 - Identify all the functional requirements paths that were made by the users of the website;

- Identify the most used navigation paths of the website;
- Automatic generated Traceability Matrix between functional requirements and implementation artifacts;
- **Recommendations to the Functional Requirements:**
 - * Priority of change of a requirement;
 - * Create new requirement;
 - * Delete existing requirement;
 - * Split existing Requirement;
 - * Create new requirement dependency;
 - * Remove requirement dependency.
- **Mapping Tool.** Tool to assist the task of tracing requirement coverage of a website through a high-level mapping tool that correlates the functional requirements of the website with the web pages and elements;
- **More Detailed Reports based on Web Usage.** REQAnalytics provide detailed reports that can effectively give hints/suggestions of how to improve the quality and level of service of websites. Traditionally, web analytics tools offer reports based on the web usage. However, there is a gap in the real utility and comprehensibility of the information gathered;
- **Experimental Evaluation.** Two case studies to assess the recommendations to the requirements specification provided by REQAnalytics, the Recommender System developed during this research work.

8.3 Opportunities for Future Research Work

The analysis over the previous research contributions allowed the author to identify several opportunities for future research arising from this work. These opportunities are:

- **Use other Web Mining methods and techniques.** Since there are emerging new methods and techniques applied to web usage mining, an opportunity for future work would be apply other Web Mining methods to achieve new type of recommendations to the requirements specification.

- **Develop other evaluation case studies.** We intend to develop other case studies on more complex websites to study other kind of recommendations.
- **Automatically separate different roles.** Using the web usage data to identify user roles could help to improve and personalize the recommendations to the software requirements specification.
- **Allow REQAnalytics to work with different Web Analytics tools.** Since each of these Web Analytics tools can provide different kinds of data and metrics to analyze and since each website uses its tool, supporting different tools would be an important improvement to REQAnalytics.
- **Study how to automatically apply the recommendations to requirements specification.** An interesting improvement to REQAnalytics would be to automatically apply the suggested recommendations to the software specification requirements in order to automatize the process of requirements management.
- **Integrate REQAnalytics with Requirements Management Tools.** Add the capabilities of the REQAnalytics along with the features of a traditional requirements management tool can be an interesting approach to research.
- **Extend the tool to analyze dependencies between requirements which belong to different sessions.** Analyse dependencies of different sessions would be an interesting research to allow to identify new requirements dependencies and to remove invalid dependencies.
- **Apply this approach to other type of applications.** REQAnalytics can only be used in websites or web applications. Applying this same approach in mobile applications would be a promising research work.

8.4 Conclusion

Requirements Management is a challenging task for developers and software engineers while developing complex software systems. This dissertation has delivered a research study which allowed to understand the process of requirements

management and evolution of websites which allowed to identify new challenges and open issues related with this research field.

In addition, a recommender system has been proposed and its applicability in websites has been evaluated through two different case studies. Finally, this dissertation has developed an original and novel approach to support the requirements maintenance and evolution by means of suggesting recommendations to the software requirements specification taking into account the web usage data of website. The contributions that arose from this research work and documented in this dissertation have led to identify several promising directions for future research in this field.

Glossary

API Application Programming Interface

Browser Web Browser

CMM Capability Maturity Model for software

DBMS Database Management System

DOM Document Object Model

HTML HyperText Markup Language

ISO International Standards Organization

JSON JavaScript Object Notation

KPI Key Performance Indicator

Log Logfile

LSI Latent Semantic Indexing

OS Operating System

OWA Open Web Analytics

PHP Hypertext PreProcessor

RDBMS Relational Database Management System

RE Requirements Engineering

REST Representational State Transfer

RM Requirements Management

RSSE Recommender System for Software Engineering

RTM Requirements Traceability Matrix

SDLC System Development Life Cycle

SIGARRA Information System for the Aggregated Management of
Resources and Academic Records

SQL Structured Query Language

SRS Software Requirements Specification

UML Unified Modeling Language

URI Uniform Resource Identifier

URL Uniform Resource Locator

VSM Vector Space Model

Web Web World Wide Web

WWW World Wide Web

XML eXtensible Markup Language

XSD XML Schema Definition

Bibliography

- [ABB12] Rajendra Akerkar, Costin Badica, and Dumitru Dan Burdescu, *Desiderata for research in web intelligence, mining and semantics*, Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics - WIMS '12 (New York, USA), ACM Press, 2012, p. 1. Cited on pp. 3 and 48.
- [AP01] A.I. Anton and C. Potts, *Functional paleontology: system evolution as the user sees it*, Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001, IEEE Comput. Soc, 2001, pp. 421–430. Cited on pp. 25 and 27.
- [AT05] G. Adomavicius and A. Tuzhilin, *Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions*, IEEE Transactions on Knowledge and Data Engineering 17 (2005), no. 6, 734–749. Cited on p. 55.
- [Ban11] Ansuman Banerjee, *Requirement Evolution Management: A Systematic Approach*, 2011 IEEE Computer Society Annual Symposium on VLSI, IEEE, 2011, pp. 150–155. Cited on p. 4.
- [BCL11] Mark Barratt, Thomas Y. Choi, and Mei Li, *Qualitative case studies in operations management: Trends, research outcomes, and future research implications*, Journal of Operations Management 29 (2011), no. 4, 329–342. Cited on p. 106.
- [BHCoo] Chumki Basu, Haym Hirsh, and William Cohen, *Recommendation as Classification: Using Social and Content-Based Information in Recommendation*. Cited on pp. 53 and 54.
- [BJo8] Danielle Booth and Bj Jansen, *A review of methodologies for analyzing websites*, Handbook of research on web log analysis (2008), 141–162. Cited on pp. xi, 45, and 46.
- [BS97] Marko Balabanović and Yoav Shoham, *Fab: content-based, collaborative recommendation*, Communications of the ACM 40 (1997), no. 3, 66–72. Cited on pp. 52 and 54.
- [Buro2] Robin Burke, *Hybrid Recommender Systems: Survey and Experiments*, User Modeling and User-Adapted Interaction 12 (2002), no. 4, 331–370. Cited on p. 53.
- [Buro7] ———, *Hybrid web recommender systems*, 377–408. Cited on p. 53.
- [CDH10] David Cuddeback, Alex Dekhtyar, and Jane Hayes, *Automated Requirements Traceability: The Study of Human Analysts*, 2010 18th IEEE International Requirements Engineering Conference, IEEE, sep 2010, pp. 231–240. Cited on p. 33.
- [CH10] Carlos Castro-Herrera, *A hybrid recommender system for finding relevant users in open source forums*, 2010 Third International Workshop on Managing Requirements Knowledge, IEEE, sep 2010, pp. 41–50. Cited on p. 57.
- [CHCC03] J. Cleland-Huang, C.K. Chang, and M. Christensen, *Event-based traceability for managing evolutionary change*, IEEE Transactions on Software Engineering 29 (2003), no. 9, 796–810 (English). Cited on p. 29.

- [CHCH09] Carlos Castro-Herrera and Jane Cleland-Huang, *A Machine Learning Approach for Identifying Expert Stakeholders*, 2009 Second International Workshop on Managing Requirements Knowledge, IEEE, sep 2009, pp. 45–49. Cited on p. 57.
- [CHCH10] ———, *Utilizing recommender systems to support software requirements elicitation*, Proceedings of the 2nd International Workshop on Recommendation Systems for Software Engineering - RSSE '10 (New York, New York, USA), ACM Press, may 2010, pp. 6–10. Cited on p. 55.
- [CHGH⁺14] Jane Cleland-Huang, Orlena C. Z. Gotel, Jane Huffman Hayes, Patrick Mäder, and Andrea Zisman, *Software traceability: trends and future directions*, Proceedings of the on Future of Software Engineering - FOSE 2014 (New York, New York, USA), ACM Press, may 2014, pp. 55–69. Cited on p. 33.
- [CHMo8] Jane Cleland-Huang and Bamshad Mobasher, *Using data mining and recommender systems to scale up the requirements process*, Proceedings of the 2nd international workshop on Ultra-large-scale software-intensive systems - ULSSIS '08 (New York, New York, USA), ACM Press, may 2008, pp. 3–6. Cited on p. 57.
- [Chr] Chris McFadden, *Optimizing the Online Business Channel with Web Analytics*. Cited on pp. xi and 45.
- [CL96] Jim Cowie and Wendy Lehnert, *Information extraction*, Communications of the ACM 39 (1996), no. 1, 80–91. Cited on p. 40.
- [CMS99] Robert Cooley, Bamshad Mobasher, and Jaideep Srivastava, *Data Preparation for Mining World Wide Web Browsing Patterns*, Knowledge and Information Systems 1 (1999), no. 1, 5–32. Cited on pp. 42 and 43.
- [CNA⁺11] Juan M. Carrillo de Gea, Joaquín Nicolás, José L. Fernández Alemán, Ambrosio Toval, Christof Ebert, and Aurora Vizcaíno, *Requirements Engineering Tools*, IEEE Software 28 (2011), no. 4, 86–91 (English). Cited on pp. xi, 30, and 35.
- [coe] CoEST: Center of excellence for software traceability,. Cited on p. 27.
- [Com11] Committee for Advancing Software Intensive Systems Producibility (CASISP), *Critical Code: Software Producibility for Defense*, 2011. Cited on p. 28.
- [cra15] Crazyegg. Available from <http://www.crazyegg.com>. Cited on p. 46.
- [DCVP03] Daniela Damian, James Chisan, Lakshminarayanan Vaidyanathasamy, and Yogendra Pal, *An Industrial Case Study of the Impact of Requirements Engineering on Downstream Development*, 40. Cited on p. 24.
- [DGH⁺11] Horatiu Dumitru, Marek Gibiec, Negar Hariri, Jane Cleland-Huang, Bamshad Mobasher, Carlos Castro-Herrera, and Mehdi Mirakhorli, *On-demand feature recommendations derived from mining public product descriptions*, Proceeding of the 33rd international conference on Software engineering - ICSE '11 (New York, New York, USA), ACM Press, may 2011, p. 181. Cited on p. 58.
- [DL12] Antonina Danylenko and Welf Löwe, *Context-aware recommender systems for non-functional requirements*, 80–84. Cited on p. 58.
- [DM03] H Dai and Bamshad Mobasher, *A road map to more effective web personalization: Integrating domain knowledge with web usage mining*, Proceedings Of The International Conference On Internet Computing, vol. 2003, 2003, pp. 1–8. Cited on p. 5.
- [Dono2] Qi Dong, *Predicting and managing system interactions at early phase of the product development process*, Phd, 2002. Cited on p. 19.

- [DP98] Ralf Dömges and Klaus Pohl, *Adapting traceability environments to project-specific needs*, Communications of the ACM **41** (1998), no. 12, 54–62. Cited on p. 27.
- [DS08] Joumana Dargham and Rima Semaan, *A Navigational Web Requirements Validation through Animation*, 2008 Third International Conference on Internet and Web Applications and Services, IEEE, jun 2008, pp. 211–216. Cited on p. 26.
- [DZ09] Fangshu Di and Maolin Zhang, *An Improving Approach for Recovering Requirements-to-Design Traceability Links*, 2009 International Conference on Computational Intelligence and Software Engineering, IEEE, dec 2009, pp. 1–6. Cited on p. 28.
- [ED] C. Ebert and J. De Man, *Requirements uncertainty: influencing factors and concrete improvements*, Proceedings. 27th International Conference on Software Engineering, 2005. ICSE 2005., IEEe, pp. 553–560 (English). Cited on p. 25.
- [EG02] A. Egyed and P. Grunbacher, *Automating requirements traceability: Beyond the record & replay paradigm*, Proceedings 17th IEEE International Conference on Automated Software Engineering,, IEEE Comput. Soc, 2002, pp. 163–171 (English). Cited on p. 33.
- [EG07] Kathleen M. Eisenhardt and Melissa E. Graebner, *Theory building from cases: Opportunities and challenges*, Academy of Management Journal **50** (2007), no. 1, 25–32. Cited on p. 106.
- [EMM01] Khaled El Emam, Walcelio Melo, and Javam C. Machado, *The prediction of faulty classes using object-oriented design metrics*, Journal of Systems and Software **56** (2001), no. 1, 63–75. Cited on p. 24.
- [ER12] Michael Ekstrand and John Riedl, *When recommenders fail*, Proceedings of the sixth ACM conference on Recommender systems - RecSys '12 (New York, New York, USA), ACM Press, sep 2012, p. 233. Cited on p. 53.
- [Fin15] Anthony Finkelstein, *Advanced Software Engineering: Requirements Management*, Tech. report, 2015. Cited on p. 4.
- [FLF12] Camilo Fitzgerald, Emmanuel Letier, and Anthony Finkelstein, *Early failure prediction in feature request management systems: an extended study*, Requirements Engineering **17** (2012), no. 2, 117–132. Cited on p. 57.
- [Fo005] Food and Drug Administration (FDA), *Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices*, 2005. Cited on p. 27.
- [FRST15] Alexander Felfernig, Stefan Reiterer, Martin Stettinger, and Juha Tiihonen, *Intelligent Techniques for Configuration Knowledge Evolution*, Proceedings of the Ninth International Workshop on Variability Modelling of Software-intensive Systems, VaMoS '15, Hildesheim, Germany, January 21-23, 2015, 2015, p. 51. Cited on pp. 38 and 58.
- [Geo] Paul Krause George Spanoudakis, Andrea Zisman, Elena Pérez-miñana, *Rule-Based Generation of Requirements Traceability Relations*. Cited on p. 28.
- [GF94] O.C.Z. Gotel and C.W. Finkelstein, *An analysis of the requirements traceability problem*, Proceedings of IEEE International Conference on Requirements Engineering, IEEE Comput. Soc. Press, 1994, pp. 94–101. Cited on pp. 27, 28, and 33.
- [GFD⁺14] Florent Garcin, Boi Faltings, Olivier Donatsch, Ayar Alazzawi, Christophe Bruttin, and Amr Huber, *Offline and online evaluation of news recommender systems at swiss-info.ch*, Proceedings of the 8th ACM Conference on Recommender systems - RecSys '14 (New York, New York, USA), ACM Press, oct 2014, pp. 169–176. Cited on p. 52.

- [GGF14] Florent Garcin, Frederik Galle, and Boi Faltings, *Focal*, Proceedings of the 8th ACM Conference on Recommender systems - RecSys '14 (New York, New York, USA), ACM Press, oct 2014, pp. 369–370. Cited on p. 53.
- [GKV14] Arda Goknil, Ivan Kurtev, and Klaas Van Den Berg, *Generation and validation of traces between requirements and architecture based on formal trace semantics*, Journal of Systems and Software **88** (2014), 112–137. Cited on p. 52.
- [GLX⁺11] T.a Gao, T.b Li, Z.b Xie, J.a Xu, and Ya Qian, *A process model of software evolution requirement based on feedback*, Proceedings - 2011 International Conference of Information Technology, Computer Engineering and Management Sciences, ICM 2011, vol. 2, 2011, pp. 171–174. Cited on pp. 4 and 58.
- [goo15] *Google Analytics*. Available from <https://www.google.com/analytics/>, 2015. Cited on pp. 46, 78, and 94.
- [GP14] Ana Garcia and Ana C R Paiva, *SaaS Usage Information for Requirements Maintenance*, 16th International Conference on Enterprise Information Systems (ICEIS), 2014. Cited on pp. 4 and 58.
- [GP16a] Jorge Esparteiro Garcia and Ana C R Paiva, *A Requirements-to-Implementation Mapping Tool for Requirements Traceability*, Journal of Software **11** (2016), no. 2, 193–200. Cited on p. 13.
- [GP16b] ———, *REQAnalytics : A Recommender System for Requirements Maintenance*, International Journal of Software Engineering and Its Applications **10** (2016), no. 1, 129–140. Cited on p. 13.
- [GPST14] Carlo Ghezzi, Mauro Pezzè, Michele Sama, and Giordano Tamburrelli, *Mining behavior models from user-intensive web applications*, Proceedings of the 36th International Conference on Software Engineering - ICSE 2014 (New York, New York, USA), ACM Press, 2014, pp. 277–287. Cited on pp. 4 and 56.
- [GS09] Asela Gunawardana and Guy Shani, *A Survey of Accuracy Evaluation Metrics of Recommendation Tasks*, The Journal of Machine Learning Research **10** (2009), 2935–2962. Cited on p. 53.
- [HD05] Jane Huffman Hayes and Alex Dekhtyar, *Humans in the traceability loop*, Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering - TEFSE '05 (New York, New York, USA), ACM Press, nov 2005, p. 20. Cited on p. 82.
- [HDO03] J.H. Hayes, A. Dekhtyar, and J. Osborne, *Improving requirements tracing via information retrieval*, Journal of Lightwave Technology, IEEE Comput. Soc, 2003, pp. 138–147. Cited on pp. 28, 34, and 94.
- [HDS06] J.H. Hayes, A. Dekhtyar, and S.K. Sundaram, *Advancing candidate link generation for requirements tracing: the study of methods*, IEEE Transactions on Software Engineering **32** (2006), no. 1, 4–19. Cited on pp. 32 and 33.
- [HF01] Ivy F. Hooks and Kristin A. Farry, *Customer-centered Products: Creating Successful Products Through Smart Requirements Management*, 2001. Cited on pp. 19 and 26.
- [HK12] Hamed Hassanzadeh and Mohammad Reza Keyvanpour, *Semantic Web Requirements through Web Mining Techniques*, 616–620. Cited on p. 39.

- [HKL10] Youngki Hong, Minh Kim, and Sang-Woong Lee, *Requirements Management Tool with Evolving Traceability for Heterogeneous Artifacts in the Entire Life Cycle*, 2010 Eighth ACIS International Conference on Software Engineering Research, Management and Applications, IEEE, 2010, pp. 248–255. Cited on pp. 3 and 30.
- [IEE10] IEEE, *Systems and software engineering – Vocabulary*, 1–418 (English). Cited on p. 18.
- [IWD09] Noraini Ibrahim, Wan M N Wan Kadir, and Safaai Deris, *Propagating requirement change into software high level designs towards resilient software evolution*, Proceedings - Asia-Pacific Software Engineering Conference, APSEC, 2009, pp. 347–354. Cited on p. 2.
- [KBoo] Raymond Kosala and Hendrik Blockeel, *Web mining research*, ACM SIGKDD Explorations Newsletter 2 (2000), no. 1, 1–15. Cited on pp. 40 and 42.
- [KKKC08] Vassilka Kirova, Neil Kirby, Darshak Kothari, and Glenda Childress, *Effective requirements traceability: Models, tools, and practices*, Bell Labs Technical Journal 12 (2008), no. 4, 143–157. Cited on p. 30.
- [KMA⁺13] Asma Khatoon, Yasir Hafeez Motla, Madiha Azeem, Humera Naz, and Sana Nazir, *Requirement change management for global software development using ontology*, 2013 IEEE 9th International Conference on Emerging Technologies (ICET), IEEE, dec 2013, pp. 1–6. Cited on pp. 4, 37, and 58.
- [KPSC07] Youngjoong Ko, Sooyong Park, Jungyun Seo, and Soonhwang Choi, *Using classification techniques for informal requirements in the requirements analysis-supporting system*, Information and Software Technology 49 (2007), no. 11-12, 1128–1140. Cited on p. 57.
- [KS09] Andrew Kannenberg and Dr. Hossein Saiedian, *Why Software Requirements Traceability Remains a Challenge*, 2009. Cited on pp. xv, 3, 27, 28, 30, 33, 37, and 63.
- [KSK12] Lakhwinder Kumar, Hardeep Singh, and Ramandeep Kaur, *Web analytics and metrics*, Proceedings of the International Conference on Advances in Computing, Communications and Informatics - ICACCI '12 (New York, New York, USA), ACM Press, aug 2012, p. 966. Cited on pp. 2, 3, 5, 40, 46, 47, 48, and 71.
- [LADR12] João Lemos, Carina Alves, Leticia Duboc, and Genaina Nunes Rodrigues, *A systematic mapping study on creativity in requirements engineering*, Proceedings of the 27th Annual ACM Symposium on Applied Computing - SAC '12 (New York, New York, USA), ACM Press, mar 2012, p. 1083. Cited on p. 59.
- [LFVV11] I. Lopes Margarido, J.P. Faria, R.M. Vidal, and M. Vieira, *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, 2011, pp. 1–6. Cited on p. 63.
- [Liu11] Bing Liu, *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*, Springer Science & Business Media, 2011. Cited on p. 53.
- [LM06] D. Lempia and S. Miller, *Requirements engineering management*, National Software and Complex Electronic Hardware Standardization Conference (Atlanta, GA), 2006. Cited on p. 28.
- [LM12] Yang Li and Walid Maalej, *Requirements Engineering: Foundation for Software Quality*, Lecture Notes in Computer Science, vol. 7195, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Cited on p. 34.
- [Low99] David Lowe, *Hypermedia and the Web: An Engineering Approach*. Cited on pp. 18 and 23.

- [LQF10] Soo Ling Lim, Daniele Quercia, and Anthony Finkelstein, *StakeNet*, Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - ICSE '10 (New York, New York, USA), vol. 1, ACM Press, 2010, p. 295. Cited on pp. 25 and 57.
- [LvDNdZ04] M. Lormans, A. van Deursen, E. Nocker, and A. de Zeeuw, *Managing evolving requirements in an outsourcing context: an industrial experience report*, Proceedings. 7th International Workshop on Principles of Software Evolution, 2004., IEEE, 2004, pp. 149–158. Cited on p. 25.
- [MA15] Bhupendra Kumar Malviya and Jitendra Agrawal, *A Study on Web Usage Mining Theory and Applications*, 2015 Fifth International Conference on Communication Systems and Network Technologies, IEEE, apr 2015, pp. 935–939. Cited on pp. 2 and 41.
- [MJZCH13] Patrick Mader, Paul L. Jones, Yi Zhang, and Jane Cleland-Huang, *Strategic Traceability for Safety-Critical Projects*, IEEE Software 30 (2013), no. 3, 58–66. Cited on p. 4.
- [MRE12] Jamshaid G. Mohebzada, Guenther Ruhe, and Armin Eberlein, *Systematic mapping of recommendation systems for requirements engineering*, 200–209. Cited on pp. xi, 55, 56, 57, and 59.
- [MT09] Walid Maalej and Anil Kumar Thurimella, *Towards a Research Agenda for Recommendation Systems in Requirements Engineering*, 2009 Second International Workshop on Managing Requirements Knowledge, IEEE, sep 2009, pp. 32–39. Cited on pp. 52 and 57.
- [MT13] Walid Maalej and Anil Kumar Thurimella (eds.), *Managing Requirements Knowledge*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. Cited on pp. 5, 48, 58, 59, and 65.
- [NE00] Bashar Nuseibeh and Steve Easterbrook, *Requirements engineering: a roadmap*, Proceedings of the conference on The future of Software engineering - ICSE '00 (New York, New York, USA), ACM Press, may 2000, pp. 35–46. Cited on pp. 24 and 30.
- [NG06] Christian Neumuller and Paul Grunbacher, *Automating Software Traceability in Very Small Companies: A Case Study and Lessons Learne*, 21st IEEE/ACM International Conference on Automated Software Engineering (ASE'06), IEEE, 2006, pp. 145–156 (English). Cited on p. 33.
- [PB07] Michael J. Pazzani and Daniel Billsus, *Content-based recommendation systems*, 325–341. Cited on pp. 52 and 56.
- [PFTV05] Ana C. R. Paiva, João C. P. Faria, Nikolai Tillmann, and Raul A. M. Vidal, *A model-to-implementation mapping tool for automated model-based GUI testing*, ICFEM'05 Proceedings of the 7th international conference on Formal Methods and Software Engineering (Berlin, Heidelberg) (Kung-Kiu Lau and Richard Banach, eds.), Lecture Notes in Computer Science, vol. 3785, Springer Berlin Heidelberg, nov 2005, pp. 450–464. Cited on p. 79.
- [piw15] *Piwik - Open Analytics Platform*. Available from <http://piwik.org/>, 2015. Cited on pp. 46, 78, and 94.
- [Poh96] Klaus Pohl, *Process-Centered Requirements Engineering*, John Wiley & Sons, Inc., nov 1996. Cited on p. 30.
- [Poh10] ———, *Requirements Engineering: Fundamentals, Principles, and Techniques*, Springer Publishing Company, Incorporated, jul 2010. Cited on pp. 18, 20, and 22.

- [PRRS13] Deepak Pai, Balaraman Ravindran, Shyam Rajagopalan, and Ramesh Srinivasaraghavan, *Automated faceted reporting for web analytics*, Proceedings of the 4th international workshop on Web-scale knowledge representation retrieval and reasoning - Web-KR '13 (New York, New York, USA), ACM Press, nov 2013, pp. 9–16. Cited on pp. 3, 48, and 71.
- [RJ01] B. Ramesh and M. Jarke, *Toward reference models for requirements traceability*, IEEE Transactions on Software Engineering 27 (2001), no. 1, 58–93. Cited on p. 27.
- [RMWZ14] Martin P. Robillard, Walid Maalej, Robert J. Walker, and Thomas Zimmermann (eds.), *Recommendation Systems in Software Engineering*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. Cited on p. 59.
- [RMZR08] Jose Romero-Mariona, Hadar Ziv, and Debra J. Richardson, *SRRS: a recommendation system for security requirements*, Proceedings of the 2008 international workshop on Recommendation systems for software engineering - RSSE '08 (New York, New York, USA), ACM Press, nov 2008, p. 50. Cited on p. 57.
- [Roy90] W. Royce, *TRW's Ada process model for incremental development of large software systems*, 2–11. Cited on pp. 22 and 26.
- [RR06] Suzanne Robertson and James Robertson, *Mastering the Requirements Process (2nd Edition)*. Cited on p. 21.
- [RWZ10] Martin Robillard, Robert Walker, and Thomas Zimmermann, *Recommendation Systems for Software Engineering*, IEEE Software 27 (2010), no. 4, 80–86. Cited on pp. 51, 55, 56, 58, and 59.
- [SA03] Susanne A. Sherba and Kenneth M. Anderson, *A Framework for Managing Traceability Relationships Between Requirements And Architectures*. Cited on pp. 30, 32, and 33.
- [SAW13] Bahar Sateli, Elian Angius, and Rene Witte, *The ReqWiki Approach for Collaborative Software Requirements Engineering with Integrated Text Analysis Support*, 2013 IEEE 37th Annual Computer Software and Applications Conference, IEEE, jul 2013, pp. 405–414. Cited on p. 25.
- [Shr12] Aditi Shrivastava, *Clustering of Web Log Data to Analyze User Navigation*, International Journal of Scientific and Research Publications 2 (2012), no. 2, 1–4. Cited on pp. 42 and 43.
- [SK] K. R. Suneetha and R. Krishnamoorthi, *Identifying User Behavior by Analyzing Web Server Access Log File*, no. 4. Cited on p. 42.
- [SK98] Ian Sommerville and Gerald Kotonya, *Requirements Engineering: Processes and Techniques*. Cited on pp. 22 and 25.
- [SKM11] Hossein Saiedian, Andrew Kannenberg, and Serhiy Morozov, *A streamlined, cost-effective database approach to manage requirements traceability*, Software Quality Journal 21 (2011), no. 1, 23–38. Cited on pp. 32 and 33.
- [SKS14] Himani Singal, Shruti Kohli, and Amit Kumar Sharma, *Web analytics: State-of-art & literature assessment*, 2014 5th International Conference - Confluence The Next Generation Information Technology Summit (Confluence), IEEE, sep 2014, pp. 24–29. Cited on pp. 3, 39, 45, 47, and 48.
- [Som04] Ian Sommerville, *Software Engineering*, Addison Wesley, 7th ed., 2004. Cited on pp. 24 and 25.
- [Som07] ———, *Software engineering*, 2007, p. 865. Cited on p. 20.

- [SOSA99] George E. Stark, Paul Oman, Alan Skillicorn, and Alan Ameele, *An examination of the effects of requirements changes on software maintenance releases*, *Journal of Software Maintenance: Research and Practice* **11** (1999), no. 5, 293–309. Cited on p. 27.
- [SPUP02] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock, *Methods and metrics for cold-start recommendations*, *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '02* (New York, New York, USA), ACM Press, aug 2002, p. 253. Cited on p. 54.
- [SS10] Brijendra Singh and Hemant Kumar Singh, *Web Data Mining research: A survey*, 2010 IEEE International Conference on Computational Intelligence and Computing Research, IEEE, dec 2010, pp. 1–10. Cited on pp. 1 and 40.
- [Sta95] Standish Group, *CHAOS Report*, Group (1995), 11–13. Cited on pp. 2 and 63.
- [TKG07] Nima Taghipour, Ahmad Kardan, and Saeed Shiry Ghidary, *Usage-based web recommendations*, *Proceedings of the 2007 ACM conference on Recommender systems - RecSys '07* (New York, New York, USA), ACM Press, oct 2007, p. 113. Cited on p. 1.
- [TWF⁺15] Ryosuke TSUCHIYA, Hironori WASHIZAKI, Yoshiaki FUKAZAWA, Tadahisa KATO, Masumi KAWAKAMI, and Kentaro YOSHIMURA, *Recovering Traceability Links between Requirements and Source Code Using the Configuration Management Log*, apr 2015, pp. 852–862. Cited on p. 28.
- [Ver11] Vikas Verma, *Comprehensive Survey of Framework for Web Personalization using Web Mining*, *International Journal of Computer Applications* **35** (2011), no. 3. Cited on pp. 3 and 48.
- [VMKR13] Chintan R. Varnagar, Nirali N. Madhak, Trupti M. Kodinariya, and Jayesh N. Rathod, *Web usage mining: A review on process, methods and techniques*, 2013 International Conference on Information Communication and Embedded Systems (ICICES), IEEE, feb 2013, pp. 40–46. Cited on p. 44.
- [VP11] Pedro Valderas and Vicente Pelechano, *A Survey of Requirements Specification in Model-Driven Development of Web Applications*, *ACM Transactions on the Web* **5** (2011), no. 2, 1–51. Cited on pp. 4 and 57.
- [Wes06] Linda Westfall, *Bidirectional Requirements Traceability*, Tech. report, The Westall Team, 2006. Cited on pp. xi, 29, and 32.
- [Wie99] Karl Eugene Wiegers, *Automating Requirements Management*, 1999. Cited on pp. 2 and 63.
- [Wie03] ———, *Software Requirements*, 3rd ed., Microsoft Press, apr 2003. Cited on pp. xi, 3, 20, 25, and 29.
- [Wil11] Craig L Williams, *A Many-to-Many (M:N) Relation Model to improve Requirements Traceability*, 2011. Cited on pp. 26 and 28.
- [woo15] Woopra. Available from <http://www.woopra.com/>, 2015. Cited on p. 78.
- [YF05] Yunwen Ye and Gerhard Fischer, *Reuse-Conducive Development Environments*, *Automated Software Engineering* **12** (2005), no. 2, 199–235. Cited on pp. 55 and 57.
- [Yin09] Robert K. Yin, *Case Study Research: Design and Methods*, SAGE Publications, 2009. Cited on pp. xii and 106.

- [Yun] Edward I. George Yung-hsin Chen, *A Bayesian Model for Collaborative Filtering*. Cited on p. 54.
- [Zav97] Pamela Zave, *Classification of research efforts in requirements engineering*, *ACM Computing Surveys* 29 (1997), no. 4, 315–321. Cited on p. 22.
- [ZSo8] Qingyu Zhang and Richard S. Segall, *Web Mining: A survey of current research, technoques, and software*, *International Journal of Information Technology & Decision Making* 07 (2008), no. 04, 683–720 (en). Cited on p. 40.
- [ZW98] M.V. Zelkowitz and D.R. Wallace, *Experimental models for validating technology*, *Computer* 31 (1998), no. 5, 23–31 (English). Cited on p. 7.