MESG

MESTRADO EM ENGENHARIA DE SERVIÇOS E GESTÃO

Requirements management based on the usage of a service

Anca-Maria Bîzoi

Master Thesis

Supervisor at FEUP: Prof. Ana Cristina Ramada Paiva (PhD)

Co-Supervisor: Prof. Jorge Manuel Esparteiro Garcia (PhD)

U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

2018-06-26

Abstract

According to recent studies, the four billion Internet users in the world, representing 54.4% of the world population, have access to more than 171 million active websites. By using specialized tools, website owners can capture data related to all the users' interactions with their website.

In the context of electronic services, by analyzing this data, relevant knowledge is gathered that can help the service providers in improving the user experience with the service, identifying new development opportunities, observing user navigation patterns, predicting user behavior, or a better understanding of the service usage.

Any improvement or alteration of the service functions involves changing the initial requirements, therefore, the necessity of a proper requirements management. Garcia and Paiva (2016a) presented an innovative approach in web-based electronic services requirements management, that involves mapping the requirements of the service with their website implementation, as for example, their equivalent URLs (Uniform Resource Locator), and analysing the way the user interacts with them. They presented REQAnalytics, a recommender system that collects the information about the usage of a website, processes it, and generates recommendations to the requirements specification of the website, such as requirement prioritization, functionalities to create or remove, and traceability between website elements and functionalities.

Given the challenges that still exist in the requirements management activities, corroborated with the potential of improvement in the existing functionalities, this research work proposes an evolution of the REQAnalytics tool.

The evolution mainly involves creating two new features: Frequent Path Pattern Report feature and Goals Reports. These features generate the most frequent subsequences of performed functionalities, information about the number of sessions where a goal was achieved, number of steps taken before reaching a goal and their frequency, the shortest sessions taken to reach a goal, the longest number of steps taken in a session to reach a goal, as well as the most frequent patterns that contain the goal.

The evolution also involves refining the data provided for Software Requirements Specification, to identify if it is possible to improve the recommendations to change requirements' priorities and create new requirements

Therefore, extensions and updates of the recommendations and reports generated by the REQAnalytics recommender system are done in this research work, with the purpose of helping the requirements engineers in the requirement maintenance activities, and to improve the overall quality of the services. A case study is used to validate the proposed extensions.

Acknowledgments

Firstly, I would like to thank my supervisor, Prof. Ana Paiva, for all the support provided during these months and for understanding my way of doing things, and to Prof. Jorge Garcia, for his availability and for helping me understand the REQAnalytics tools. Also, I would like to thank Prof. Lia Patricio for all the goodwill she showed during my studies at FEUP.

Secondly, I would like to thank Prof. Theodor Borangiu and Prof. Silvia Anton from the University "Politehnica" of Bucharest for all their efforts in helping me get the scholarship and in assisting me throughout the admission process, and for all their moral support provided across this period abroad.

I would like to thank my work colleagues, Razvan and Bogdan, for being receptive to my desire to study abroad and for guiding me in my professional and personal development throughout all the years we have been working together. And to Rares, who was very kind to help me in this research work.

Finally, I would like to thank my parents for all the sacrifices they did so I can have a proper education and for their infinite love. Also, I thank Petru, my only Romanian friend in Porto, who was always there to lift my spirits, especially while working on this research work.

And a last special mention to the ERASMUS + programme, for providing the scholarship.

Table of Contents

List of Tables

List of Figures

List of abbreviations

RE – Requirements Engineering

URL – Uniform Resource Locator

WCM – Web Content Mining

WSM – Web Structure Mining

WUM – Web Usage Mining

e-Service – electronic service

HTML – HyperText Markup Language

XML – Extensible Markup Language

DOM – Document Object Model

OWA – Open Web Analytics

FEUP – Faculdade de Engenharia da Universidade do Porto

1 Introduction

According to recent studies (Internet World Stats 2017), (Netcraft 2018), the four billion Internet users in the world, representing 54.4% of the world population, have access to more than 171 million active websites. By using specialised tools, website owners can capture data related to all the users' interactions with their website.

In the context of electronic services, by analysing this data, relevant knowledge is gathered that can help the service providers in improving the user experience with the service, identifying new development opportunities, observing user navigation patterns, predicting user behaviour, or getting better understanding of the service usage.

However, in the context of requirements engineering, this data can provide useful knowledge necessary for assisting the requirements management activities and improve the overall quality of the service provided (Garcia & Paiva 2016a).

This chapter presents the project background, the research questions that this research work aims to answer, as well as the overall structure of the report.

1.1 Project background

Available literature provides numerous examples of service failures. These are a reminder that while our IT systems may be working well today, we do not have a guarantee that they will work well tomorrow. These examples show cases of organizations suffering significant operational impact through problems with the information technology and, in many cases, their operational impact was not only inside the organization, but also to significant stakeholders of the organizations.

Change in IT involves change in business system – people, process, structure, technology. If too many changes are happening all at once, the organization become bogged down in dealing with change and sacrificing other aspects of its performance in an attempt to absorb all the changes required. Sometimes, part of these changes may have an impact on the electronic services that the organization is providing.

The IT services need to have high availability, function as expected, perform in the requested parameters, implement security, protection and recovery measures. They must be maintained, customized, improved or withdrawn whenever it is required. Even if an IT service is delivered successfully, it can still be put aside if the demand for it no longer exists or is reduced (it does not bring business value, or it does not serve to its purpose anymore).

Any improvement or alteration of the service functions involves changing the initial requirements, therefore, the necessity of a proper requirements management.

In addition to this, as seen on "Chaos Report" published in 1995 (Standish Group, 1995) incomplete requirements is the most frequent reason for cancelling projects, and the second most frequent factor that can cause projects to be inadequately delivered (project is executed, but the initial allocated budget and implementation period was exceeded, and fewer working functions than initially requested). Continuous changing of requirements and specifications is found to be the sixth most frequent reason for projects failures and the third factor that can cause the projects to be inadequately delivered.

As these factors have a great influence on the project's successful implementation, it is of the utmost necessity to give more importance to the requirements engineering activities, especially elicitation and management.

Given the increase of Internet usage and of active websites, it is understood that the amount of data collected regarding the websites usage has increased. With the help of Web Mining, service providers can get a better understanding on how their services are performing and how they are used by the end-users.

Web mining can be used in e-Commerce to identify new potential customers, retain existing customers, predicts customer purchase behavior, desires and needs, provide personalized services, create more effective Internet business processes or to re-design the websites with the scope of improving usability and user perception.

Therefore, since any information related to consumer behavior has an important value in the highly competitive nature of the e-Commerce market (Wei et al. 2015), web mining has become a powerful tool that companies can use to gain competitive advantage and increase business profit.

1.2 Problem Description

Every new service that is implemented comes from a necessity, like, for example, to solve a customer's certain problem, or to reach a company's business goals. For the development process to begin, the stakeholders must express the desired capabilities, features, demands regarding the new service.

In a perfect scenario, after the requirements are elicited, validated and implemented, the service is deployed and functioning properly until the end of its life. Unfortunately, this rarely happens, due to requirement changes that may appear anytime in the service's lifecycle, from the first day of its deployment.

Depending on the request changes, requirements may be added, updated or eliminated. Implementing these changes imply additional allocation of resources in terms of time, budget, or workforce, therefore a proper requirements management is imposed.

All these changes have different causes like the appearance of new stakeholders, laws, regulations or business processes. In addition to this, by analysing the data regarding the users' interaction with the service, new requirements may emerge, as well as recommendations of improving the existing requirements implementations.

The existing tools on the market used to support the requirement management activities do not offer such a capability, which can provide useful knowledge in how requirements' implementations perform in the context of the service.

As presented by Garcia and Paiva (2016b), analyzing the usage of a website can help identify improvements and help to maintain the website and its software requirements. They introduce REQAnalytics, a recommender system that collects the information about the usage of a website, processes it, and generates reports in a user-friendly language. The reports may show the most common requirements paths in a session, recommendations of reprioritizing requirements, creating new requirements or deleting requirements.

By analysing the user sessions and the corresponding user activity (clicks) the tool detects which are the most frequent sequences of functionalities (paths) performed throughout the entire sessions. These paths are deducted from the most frequent navigation behaviours' of the

user throughout the website, from the beginning to the end of the user session. This functionality only shows the paths and the number of sessions which have performed the same sequence through the web site. Although this functionality is useful for identifying how many times the users follow the exact same workflow, it still does not reflect the patterns which are the most popular, in all the sessions.

Based on the number of accesses, the tool also provides reports that suggest reprioritization of the requirements, creation of new requirements or proposals of deleting the requirements. However, the new requirement priorities are computed based on a fixed value, which is difficult to estimate in the context of analysing a new service, where prior traffic data is not available. In addition to this, the recommendation of new requirements report is not providing enough data to support the decision of creating a new requirement.

Given the challenges that still exist in the requirements management activities, corroborated with the potential of improvement in the existing functionalities, this research work proposes an evolution of the REQAnalytics tool, by extending the analysis performed over the sequences of functionalities (Requirements) and refining the data provided for Software Requirements Specification.

This research work aims to show how this evolution can help the requirements engineers in the requirement maintenance activities, and to improve the overall quality of the services.

1.3 Research Questions

The research questions of the project include:

RQ1: Is it possible to use web usage mining for requirements management activities, with the help of REQAnalytics recommender system?

RQ2: Are the most frequent subsequences of performed functionalities, generated with the REQAnalytics recommender system, useful for the requirements management activities?

RQ3: Are the most frequent subsequences that contain a given goal, generated with the REQAnalytics recommender system, useful for the requirements management activities?

RQ4: Is the most frequent subsequence's length performed before reaching a goal, generated with the REQAnalytics recommender system, useful for the requirements management activities?

1.4 Report outline

This master thesis is organized in five different chapters.

The first chapter provides a brief introduction to the main research area of this thesis, the problem and the motivation, the main goals, the research questions, as well as the structure of the thesis.

The second chapter presents the theoretical framework and State of the Art information regarding the main concepts which are most frequently used in this project, namely Requirements Engineering, from defining requirements to addressing the change in requirements management, and Web Mining, from describing the data mining concept to analysing the methods used in Web Usage Mining.

The third chapter presents the recommender system (REQAnalytics) which was used in the case study to answer the research questions formulated in the first chapter, along with the extensions developed in this research work.

The fourth chapter shows a comparative analysis of existing approaches, the reasons for the choice of adopted approach, and how the method was used in this research work.

The fifth chapter goes into further detail about the case study steps, presented in the previous chapter.

The last chapter summarizes the results of the research work and shows how they contributed to respond to the initial research objectives.

2 Methodology

2.1 Comparative analysis of existing approaches and reasons for the choice of adopted approach

In our everyday life, we may come across the following statements: “If we do this, then that will happen”. The source of knowledge in this situation can be different, like legislation, rationale (logical), mystical sources (dreams, prophecies), intuition, senses, common perception or empirical research. The latter source is comprised of extracting necessary data and analysing it, both steps being realized with rigorosity such that the knowledge extracted is considered to be trustworthy.

The 2015 Frascati Manual (OECD 2015) defines the research and experimental development creative and systematic work undertaken in order to increase the stock of knowledge – including knowledge of humankind, culture and society – and to devise new applications of available knowledge.

According to Creswell (Creswell, 2013), a research problem is a problem or issue that leads to the need for a study. As there are diverse types of research problems, specific approaches need to be used: quantitative, qualitative, or mixed methods.

In quantitative research, numbers are used to explain findings (Kowalczyk, 2016). Advanced statistics procedures are used to analyze the data and identify relationships between different variables in order to test the initial hypothesis. Among the advantages of using quantitative research, one may find the reduction of bias and the possibility to generalize the findings to larger populations.

Unlike the previous approach, the qualitative research does not count on measures. Its aim is to understand the meaning of human action by describing the inherent or essential characteristics of social objects or human experience (Denzin & Lincoln, cited in Jackson, Drummond & Camara 2007). In this approach, data is typically collected in the participant’s setting, data analysis is inductively built from particular to general themes, and the researcher makes interpretations of the meaning of the data (Creswell & Creswell 2017).

Mixed methods combine the quantitative and qualitative approaches to provide a better understanding of the research problem.

Table 1 presents the most frequently used strategies of inquiry in each approach. (Creswell 2003)

Table 1 – Research approaches strategies of inquiry (Creswell 2013)

Quantitative	Qualitative	Mixed methods
- Experimental designs - Non-experimental designs (surveys)	- Ethnographies - Narratives - Phenomologies - Grounded theory - Case studies	- Sequential - Concurrent - Transformative

Table 2 provides a comparison of the approaches on different dimensions, excluding the data types and strategies of inquiry, since they were presented before (Teddlie and Tashakkori 2009).

Table 2 – Comparison of research approaches (Teddlie & Tashakkori 2009)

Dimension	Quantitative	Qualitative	Mixed methods
Paradigms	Postpositivism Positivism	Constructivism (and variants)	Pragmatism, transformative perspective
Role of theory, logic	Rooted in conceptual framework or theory; Hypothetico-deductive model	Grounded theory, inductive logic	Both inductive and deductive logic; inductive-deductive research cycle
Sampling	Mostly probability	Mostly purposive	Probability, purposive and mixed
Data analysis	Statistical analyses: descriptive and inferential	Thematic strategies: categorical and contextualizing	Integration of thematic and statistical; data conversion
Validity/trust worthiness issues	Internal and external validity	Trustworthiness; credibility, transferability	Inference quality; Inference transferability

As mentioned in the first chapter of this thesis, the REQAnalytics recommender system was developed to address the concept of managing requirements with the help of knowledge collected from web usage data. As there are few researches done on this subject, a qualitative approach is chosen for this research work. This approach permits more flexibility, subjectivism, creativity, and, consequently, innovation.

2.2 Method used in the project

The qualitative method which was applied in this research work is the single-case study method. Collis and Hussey (2009) defined the case study as a methodology that is used to explore a single phenomenon in a natural setting using a variety of methods to obtain in-depth knowledge.

Case studies can be used for analytic generalization (Yin 1994), also referred to as theoretical elaboration, in which the researcher uses a particular set of circumstances, like a case, as evidence to refine, dispute, support or detail a concept, model, or theory (Jackson, Drummond & Camara 2007).

Some of the advantages of using case studies is that it provides a deeper understanding of a certain problem, and it may bring to surface new research opportunities, that can be addressed in future work.

Due to its nearly universal acceptance, Yin's six-stage case study process is adopted in this research, as presented next.

1. **Plan**

In this stage, recent relevant literature was reviewed and, based on the literature gaps, the problems and objectives were identified. After this, the research questions were formulated and the research method to be used was identified.

2. **Design**

In this stage, the necessary data that needs to be collected was identified. As the REQAnalytics system uses a web analytics tool, it was important to identify a website that has relevant user activity and whose owners are open to sharing their web usage data for the purpose of this thesis. However, gaining access to and organization's website usage is a challenging task.

The initial proposal was to collect web usage data from a public institution's website or internal web applications. However, because of security reasons, this proposal did not succeed and the focus was shifted towards the private sector. Thus, an e-Commerce service that matched the demands was identified.

3. **Prepare**

In this step, the web analytics tool was configured to gather the information from service usage information. After this activity, the service provider performed the necessary configurations of the service such that the data can be collected by the web analytics tool.

4. **Collect**

For one month, the user interaction with the e-Commerce service, was collected and stored in a database hosted by FEUP. The recommender system was used in this period and, as the data records kept on growing, it was identified that some functionalities stopped working, as they were timing out (the web pages took too long to respond). As one of the functionalities was addressed in the initial research questions, it was mandatory to solve the problem. Therefore, in the development efforts to optimize the code, new research opportunities were identified, which lead to the creation of new research questions.

5. **Analyse**

In this stage, the collected data was analyzed using the REQAnalytics recommender system, which was updated with the new functionalities that were developed to support the research questions. Detailed information regarding this step is provided in the next chapter.

6. **Share**

In this final stage, the knowledge extracted from the previous step is embedded in this thesis and the conclusions are formulated.

3 Literature Review

3.1 Requirements

Different definitions of the term “requirement” are provided by internationally recognized standards:

1. statement that translates or expresses a need and its associated constraints and conditions (ISO/IEC/IEEE 12207:2017)
2. condition or capability that must be met or possessed by a system, system component, product, or service to satisfy an agreement, standard, specification, or other formally imposed documents (IEEE 730:2014)
3. provision that contains criteria to be fulfilled (ISO/IEC 14143-2:2011)
4. need or expectation that is stated, generally implied, or obligatory (ISO/IEC 19770-1:2017)

In general, the requirements can be classified in functional requirements, non-functional requirements (quality) and constraints.

Since the classification of the requirements is performed manually by the requirements engineers, being a time-consuming task, different tools for automatic requirements classification, using machine learning algorithms, have been developed. Hayes, Li and Rahimi (2014) presented a tool that can classify the requirements in functional or non-functional requirements, or even temporal or non-temporal requirements. Similarly, for classifying functional and non-functional requirements in agile development, Sunner and Bajaj (2016) propose an automatic approach, that uses text mining and classification by clustering and supervised learning through neural network with a genetic algorithm.

To solve the major problems that reviewers encounter when finding consistency or completeness defects among requirements and related information, given the complex and voluminous specification documents, Ott (2013) presented an automatic classification method using Naive-Bayes and Support Vector Machines.

As most of the non-functional requirements are often discovered in the implementation process, Cleland-Huang et al. (2007) introduced a novel approach based on information retrieval methods for detecting and classifying non-functional requirements from both structured requirements specifications as well as from free-form text. However, the results of a recent study carried by Eckhardt, Vogelsang and Fernández (2016) suggests that most non-functional requirements are not non-functional as they describe the behaviour of a system, and, consequently, should be handled similarly to functional requirements.

By training convolutional neural networks, Winkler and Vogelsang (2016) developed an automatic approach for differentiating the requirements from the auxiliary content, in a requirements specification document. The ReaCT system, presented by Dollman and Geierhos (2016) is another automated approach that achieves an accuracy of 92% in distinguishing between on- and off-topic information in the user-generated requirement descriptions.

3.2 Requirements engineering

According to Glinz (2014), requirements engineering can be defined as a systematic and disciplined approach to the specification and management of requirements with the following goals:

1. Knowing the relevant requirements, achieving a consensus among the stakeholders about these requirements, documenting them according to given standards, and managing them systematically,
2. Understanding and documenting the stakeholders' desires and needs,
3. Specifying and managing requirements to minimize the risk of delivering a system that does not meet the stakeholders' desires and needs.

Although requirements engineering has been extensively developed as a discipline, the practitioners still find it difficult to learn and apply requirements engineering as there is still no common model of Body of Knowledge (BOK) in software engineering.

First released in 2004 by the IEEE Computer Society, The Guide to the Software Engineering Body of Knowledge (SWEBOK) describes generally accepted basic concepts about 15 knowledge areas of software engineering, like, for example, software requirements, software design, software engineering management and software quality. SWEBOK is now at its third version, which has also become internationally recognized as ISO/IEC Technical Report 19759:2015.

Developed by the International Institute of Business Analysis, the Guide to the Business Analysis Body of Knowledge (BABOK) is also at its third version, released in 2015. The guide is globally recognized as a standard of practice and it organizes the most common business analysis activities in 6 knowledge areas, including elicitation and collaboration and requirements life cycle management.

In their effort to develop a new guideline for requirements engineering, called REBOK (Requirements Engineering Body Of Knowledge), Aoyama et al. (2010), after analyzing the abovementioned BOKs and other relevant literature, they identified four major challenges. These were the lack of the common role model of requirements analyst, lack of practical knowledge body, lack of common knowledge area due to diversity of the knowledge and a lack of common model of BOKs.

3.3 Requirements engineering activities

Requirements Engineering is divided in four main activities [Pohl, 2010]: elicitation, documentation, negotiation, validation and management.

Elicitation (Discovery) is the activity of seeking, obtaining, creating, interpreting, analysing and consolidating requirements from various sources, also known as stakeholders. When it comes to service design, the process of elicitation is human-centered and focuses on the usefulness and usability of the service by the customers. In this activity, the service providers may formulate requirements to improve customer satisfaction and differentiate themselves on the market.

Documentation (Specification) is the process of creating the documents that contain the requirements and the expected performance of the new service or system. The document must be written in a clear, concise and easy to understand manner, which can be used in the development part.

Validation and negotiation involve verifying whether the requirements satisfy the stakeholder's needs and agreeing on a last version of the documentation; in this process, the requirements may be reformulated, combined, re-prioritized, and even deleted, as many times as necessary. It may happen, however, for the stakeholders to have different views or understandings of the same concept or process. For example, the departments may work well independently, but not so efficiently collaborate, leading, sometimes, to a very complex design of product and service. However, the service which will be delivered must meet all the stakeholders' needs. Therefore, negotiation is needed when conflicts appear between different points of view of the stakeholders upon the service's requirements. The conflicts should be detected, expressed and resolved before the approval of the final version of the requirements specification.

Management is the activity of managing the requirements along their lifecycle, which includes updating and tracing the requirements.

As presented in a survey conducted by VersionOne (2018), agile methods are becoming more and more popular in software development activities, given the benefits they bring, like accelerated software delivery, increasing productivity or improving business/IT alignment. Since requirements engineering is an important activity of the software development process, different studies have been carried out to identify how requirements engineering is mapped on agile methodologies.

In agile requirements engineering, the requirements are elicited, analysed and specified in an ongoing and close collaboration with a customer or customer representative in order to achieve high reactivity to changes in the requirements and in the environment (Heikkilä et al. 2015)

Kassab (2014) states that some requirements engineering practices, like requirements validation, show no significant difference between agile and waterfall methods, though the responders to his study expressed greater satisfaction with regards to the efforts of requirements engineering practices applied in their agile projects compared to the waterfall projects.

As identified by Inayat et al. (2015), the challenges of agile requirements engineering are: minimal documentation, customer availability, budget and schedule estimation, inappropriate architecture finalised in earlier stages of the project, neglecting non-functional requirements, customer incompetence in terms of decision-making and complete domain knowledge, customer agreement, contractual limitations, requirements volatility, requirements change and change evaluation.

For RE to be successful in agile projects, a collaborative mindset is required among customer, product owner and the team (Gaikwad and Joeg 2017).

3.3.1 Requirements management

In a dream-scenario, after the requirements are elicited, validated and implemented, the system is deployed and functions properly until the end of its life. Unfortunately, this rarely happens, due to requirement changes that may appear anytime in the software's lifecycle, from the first day of its deployment.

All these changes have different causes and impact the software's functionality differently. Some of most common causes of requirements changes:

- New stakeholders appear in the system's context;
- Different laws, regulations, internal procedures;
- New business functions or business goals;
- Incorrect or incomplete elicitation of initial requirements;
- Emerging technologies.

Depending on the requested changes, requirements may be added, updated or eliminated. Implementing these changes imply additional allocation of resources in terms of time, budget, or workforce, therefore a proper requirements management is imposed. Figure 1 shows the major activities that are related to requirements management, according to Wiegers (2003).

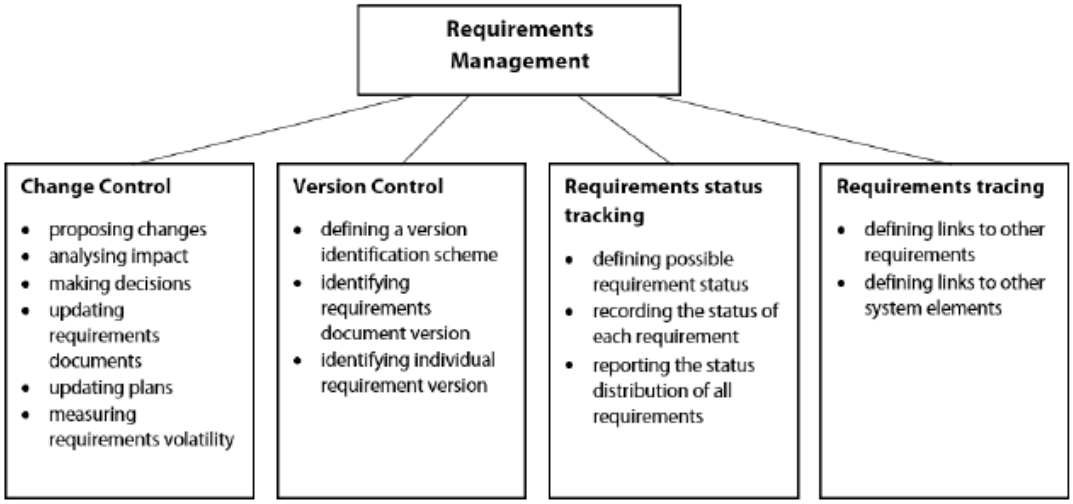

Figure 1 - Requirements management activities (Wiegers 2003)

Requirements traceability is defined by Glinz (2014), as the ability to trace a requirement (1) back to its origins, (2) forward to its implementation in design and code, (3) to requirements it depends on (and vice-versa).

Although traceability, as a common method of identifying impact of change requests, can be costly and time consuming, in most cases, the benefits are realized immediately (Jayatilleke & Lai 2018). According to a study conducted by Oliveros, Napolillo & Infesta (2016), regarding the requirements management in web applications development, requirements' change control processes have greater penetration in larger companies.

One example which proves the importance of requirements traceability is when it is identified, after some time after a service's initial deployment, that a certain functionality is not used by the users, but there are resources used for keeping that functionality in production. In this case, it is easy to observe the need to know who proposed the requirements for this functionality (the origin), to verify if they are still available or the needs have changed, and the functionality is no longer needed or needs to be deleted. If the functionality is to be deleted or

modified, then it is important to trace the requirements to requirements they depend on, to assess the impact these actions may have upon them.

Regarding the example above, if the service is implemented using web technologies, its usage and components usage or accesses can be traced using web mining and web analytics, which will be explained in the next sections.

3.4 Web mining

To understand web mining, we must first understand the concept of data mining.

As new technology continues to emerge and be used by more and more companies, users or institutions, it is understandable that there is a growth in the volume of data which is collected from different sources, like computing platforms (PCs, mobile phones, servers) or device types like sensors, security cameras or automobiles.

The McKinsey Global Institute (2016) reports that in 1986 there were only three exabytes of global data (3 billion gigabytes).

The IDC Data Age 2025 paper (2017) showed that there were 16.1 zettabytes (16.1 trillion gigabytes) of data generated in 2016. The paper also forecasts that by 2025, the global datasphere will grow to 163 zettabytes.

The data is collected to serve to a purpose, either immediately after it is collected or sometime after its capturing. Nonetheless, the increasing processing power of hardware, as well as innovation in technologies, have made the companies aware of the fact that the data they have in their repositories may be used beyond their initial purpose and transformed into a corporate asset. By mining this data, the companies can classify the user preference, identify user groups, observe patterns, predict bottlenecks or even identify new business opportunities.

Larose and Larose (2014) define datamining as the process of discovering useful patterns and trends in large data sets.

Data mining comprises the following steps: data collection, pre-processing, pattern extraction and discovery, visualization and evaluation of results.

According to the report of Internet World Stats (2017) in December 2017, there are more than four billion Internet users in the world, which account for 54.4% of the world population. Also, a Netcraft report (2018) shows that in January 2018, there were more than 171 million websites reported to be active, more than double the number of active websites reported in January 2008.

Given the increase of Internet usage and of active websites, it is understood that the amount of data collected regarding the websites usage has increased. In this context, a new branch of data mining has evolved, entitled web mining.

As defined by Liu (2011), web mining aims to discover useful information or knowledge from the web hyperlink structure, page content, and usage data.

Based on the primary kinds of data used in the mining process, web mining tasks can be categorized into three types: web structure mining, web content mining and web usage mining. Figure 2 shows the taxonomy of web mining, as presented by Singh and Kautish (2015).

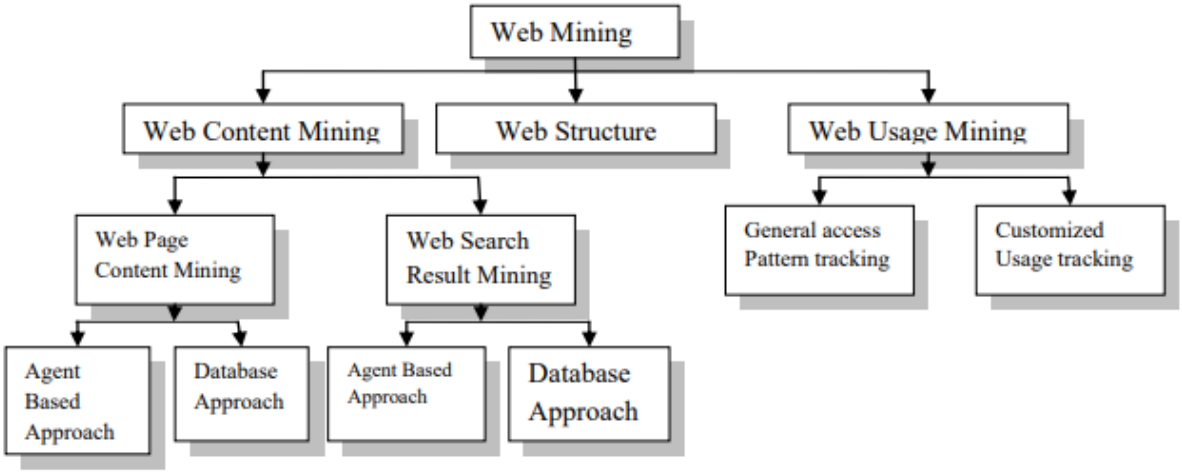

Figure 2 - Web mining taxonomy (Singh & Kautish 2015)

Web content mining (WCM) is the process of mining, processing and extracting useful information from structured or unstructured sets of data which can be found in web resources, like text, images, videos or audio. This process is mainly used for web page content mining, used for retrieving and exploring information from web pages, and search result mining, which is used for searching relevant data and indexing or organizing it with the purpose of faster knowledge discovery.

Web structure mining (WSM) is the process of mining the structure of web pages with the purpose of identifying the model of their hyperlinks and document structure and extract useful knowledge from this information. Practical applications of WSM involve categorizing webpages, identifying connections and similarities between webpages, generate overall analysis of website structure and propose new structures to improve website usability.

Web usage mining (WUM) is the process of analyzing, processing and extracting useful information from the data generated by the user interaction with the website. More information regarding WUM is presented in sub-chapter 2.5.

Search engines are a common example of how web mining can be applied to rank search results, by using different page ranking algorithms, such as Hyperlink-Induced Topic Search (HITS) algorithm, SimRank algorithm or Weighted Page Rank Algorithm.

Personalization is another example of web mining application and it implies the customization of a website's contents based on the user's needs, interests and preferences, which are deducted from its previous interactions with the service. As an example, the personalized content or user interface may include relevant advertisement, recommended products, articles or links.

Web mining can be used in e-Commerce to identify new potential customers, retain existing customers, predict customer purchase behavior, desires and needs, provide personalized services, create more effective Internet business processes or to re-design the websites with the scope of improving usability and user perception. Therefore, since any information related to consumer behavior has an important value in the highly competitive nature of the e-Commerce market (Wei et al. 2015), web mining has become a powerful tool that companies can use to gain competitive advantage and increase business profit.

Sainy and Pandey (2015) and Trappey et al. (2016) showed the usefulness of web mining of e-commerce customer reviews to compare the capabilities and prices of market competitors. Honest, Patel & Patel (2015) present a new algorithm for that analyzes user navigation patterns and identifies the page a user is more likely to access next in his session, thus showing that web pages are accessed differently, at different times, by the same users.

Pokorny and Smizanský (2005) propose a new web content mining method, entitled Page Content Rank (PCR), which uses neural network classification to calculate the page relevance ranking, according to the importance of the terms which the page contains. Pushpalatha and Reddy (2017) proposed the eXtensible Web Usage Mining Framework (XWUMF) used for processing web log data, identifying web usage patterns and discovering actionable knowledge.

Verma et al. (2015) discuss the design of a Semantic and Neural based E-commerce page ranking algorithm (SNEC) that can help customers in buying a new product, by displaying relevant web sites first, but also companies in reanalysing the structure of their website in order to gain more visibility than their competitors.

In their work, Raheja and Katiyar (2014) propose an approach for web usage mining, in which web logs are used in cluster forms, to reduce the search time of a user and to show more relevant content.

3.5 Web Usage Mining

By using specialised tools, website owners can capture data related to all the users' interactions with the service. By mining this data, relevant knowledge is gathered that can help the service providers in improving the user experience with the service, identifying new development opportunities, observing user navigation patterns, predicting user behavior, or a better understanding of the service usage.

Web usage mining is divided in three main phases: preprocessing, pattern discovery and pattern analysis (Cooley, Mobasher & Srivastava 1999).

3.5.1 Preprocessing

The first steps in web usage mining is to collect the raw data, that may give the necessary knowledge, and preprocess it. According to Bošnjak, Marić and Bošnjak (2010), most of the authors in their papers agree that data preprocessing step is the most time-consuming step in web usage analysis projects, representing 60 to 90 % of the time necessary for the completion of an entire project.

Depending on the configuration of the service, this usage data may be collected from a variety of sources like the web server logs, application server logs, cookies, proxy server logs, user sessions, user clicks and requests.

Data preprocessing involves eliminating irrelevant, incomplete, redundant, or empty records, transforming or converting data, thus obtaining a relevant and useful set of data which can be further processed in the next step.

3.5.2 Pattern discovery

In this step, different knowledge extraction and pattern discovery algorithms, which use statistics, machine learning and other data mining techniques, are applied on the set of

preprocessed data. The most commonly used methods are presented, with their usual application:

- Association rules – by computing values such as support and confidence, for example by using the Apriori algorithm, different relations or connections can be identified between webpages or other relevant data;

- Classification – by categorizing data in predetermined classes, using different algorithms like decision trees or neural networks, different patterns can be identified, for example regarding to user preferences;

- Clustering – by creating cluster objects, different usage patterns, user profiles or groups, or common user behaviours can be identified;

- Sequential patterns – by analyzing the user activity on a website, in a temporal order, patterns of pages accessed together in a session may be identified;

- Statistical analysis, which involves the analyzing means, medians or frequency of data, to improve service performance.

In this step, one or more methods can be used, separately as well as together. For example, by using association rules with classification, different class association rules are identified.

3.5.3 Pattern analysis

In this step, knowledge is extracted from the results of the previous step. The new information that is provided may be used for re-designing or re-structuring the website for improving browsing experience, personalization for higher user satisfaction and retention, overall service improvement or identifying new business opportunities.

3.6 Final considerations

Regarding the requirements engineering domain, in the studied carried out by Ambreen et al. (2018), non-functional requirements and global requirements engineering were identified as the lead emerging areas of research. Also, topics such as RE patterns, RE for small and medium enterprises and requirements ontologies received some attention lately, while requirements verification and validation lack empirical evidence.

In their research, Spoletini and Ferrari (2017) identified eliciting requirements from large set of collective data as one of the main current trends, in accordance with the current interest on big data. Part of this trend is the idea of using online store reviews as an elicitation source.

By analysing the available literature, it is easy to identify that there is a gap between web mining and requirements management. To address this issue, the REQAnalytics tool was developed (Garcia & Paiva 2016c), aiming to improve the requirements maintenance based on the usage of the website where they were implemented. More information about this tool will be presented in the chapter to follow.

4 REQAnalytics recommender system – existing functionalities and extensions

As mentioned in Chapter 1.2, the existing tools on the market used to support the requirement management activities do not offer the capability of analyzing the web usage data, which can provide useful knowledge in how requirements' implementations perform in the context of the service usage.

To research the potential this kind of tool can have in the context of requirements management, Garcia and Paiva (2016a), introduced REQAnalytics, a recommender system that collects the information about the usage of a website, processes it, and generates reports in a user-friendly language.

Given the challenges that still exist in the requirements management activities, corroborated with the potential of improvement in the existing functionalities, the research work proposes an evolution of the REQAnalytics tool, by extending the Requirements Paths and Software Requirements Specification functionalities.

The practical objectives of this research work are to:

- Identify if it is possible to use web usage mining for requirements management activities;
- Obtain the most frequent subsequences of performed functionalities;
- Identify the most frequent subsequences that contain a given goal;
- Calculate the most frequent subsequence's length performed before reaching a goal;
- Identify if it is possible to improve the recommendations to create new requirements.

The research works aims to show how this evolution can help the requirements engineers in the requirement maintenance activities, and also to improve the overall quality of the services.

This chapter presents the recommender system (REQAnalytics) which was used in the case study to answer the research questions formulated in the first chapter, along with the extensions developed in this research work.

4.1 REQAnalytics – existing functionalities

REQAnalytics is a recommender system which has the purpose to support a website's requirements maintenance process, by analysing the web usage data of the website (Garcia & Paiva, 2016b).

A functional demo of REQAnalytics system, including the extensions developed in this research work, is available at https://web.fe.up.pt/˜reqanalytics.

In order for the REQAnalytics system to generate recommendations, the following workflow is implemented:

- The functional requirements of the website are imported in the system, through an XML document, where the attributes of the requirements are defined: ID, Title, Description, Status, Owner, Date and Priority;
- The requirements are mapped on the webpage URLs or webpage elements that represent the actual implementation of the requirements;

- The open web analytics tool, described later in Section 3.1.3, is configured to collect data from the website and it generates a script which will be embedded in the website;
- The website is configured to use the script generated by the open web analytics tool, to send the web usage data to the recommender system's database;
- The recommender system analyses the requirements behaviour based on the data which was collected from the website;
- The results are presented as recommendation reports, where different improvements, updates or statistics of requirements are displayed in a user-friendly manner.

The main features of the recommender system are described in the next subsection.

4.1.1 Requirements import

The functional requirements and their attributes are saved in an XML format and imported in the system, after which they can be further processed. Figure 3 presents the implementation of this feature in the system.

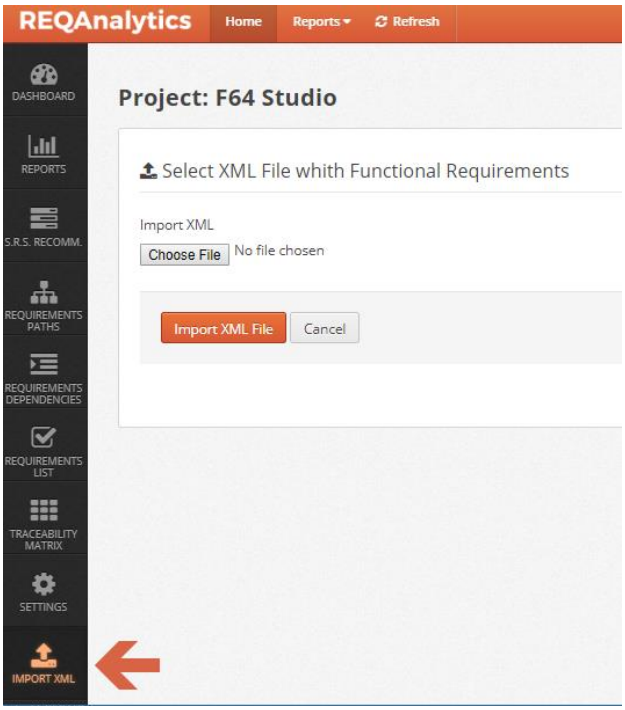

Figure 3 - Screenshot of REQAnalytics - Upload XML File with Functional Requirements

4.1.2 Mapping tool

The mapping tool is a bookmarklet which runs in the web browser and it has the purpose to create the mappings between the requirements which were imported and the website artefacts (web pages or web page elements) that implement them. Figure 4 presents the implementation of this feature in the system.

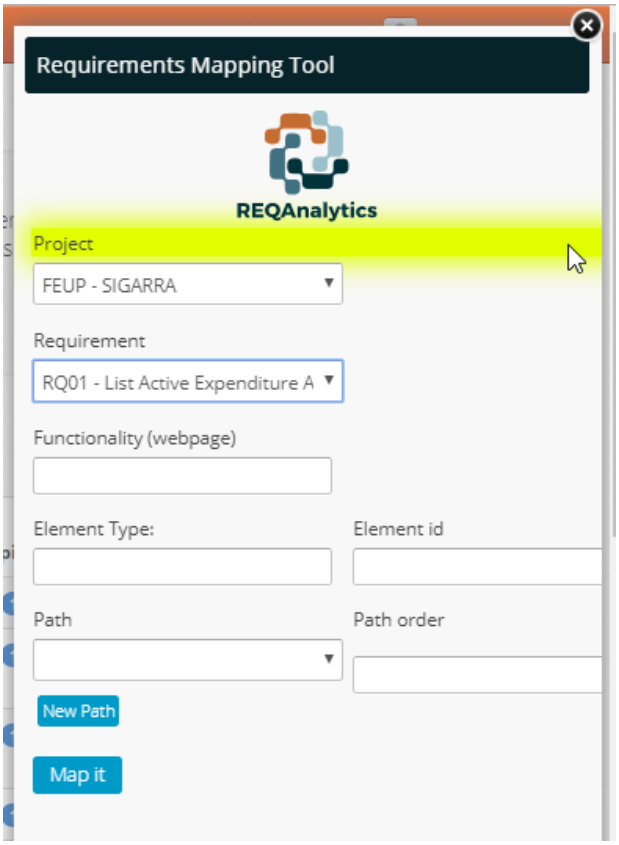

Figure 4 - Screenshot of REQAnalytics - Mapping tool

4.1.3 Integration with an open-source web analytics tool

Open Web Analytics (OWA) is the open-source web analytics tool which is used by the system to collect the user activity on the website (web usage data). To collect data about website's user activity, the OWA tool generates a JavaScript tracking code that must be inserted in the webpage's source code. The code which was used in the Case Study presented in Chapter 5 is shown in Figure 5.

```
<!-- Start Open Web Analytics Tracker -->
<script type="text/javascript">
//<![CDATA[
var owa_baseUrl = 'https://paginas.fe.up.pt/~reqanalytics/OWA1/';
var owa_cmds = owa_cmds || [];
owa_cmds.push(['setSiteId', '0ec8f56d0ce3a21c3c961a496eef866b']);
owa_cmds.push(['trackPageView']);
owa_cmds.push(['trackClicks']);

(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:') : owa_baseUrl );
    _owa.src = owa_baseUrl + 'modules/base/js/owa.tracker-combined-min.js';
    var _owa_s = document.getElementsByTagName('script')[0]; _owa_s.parentNode.insertBefore(
        _owa, _owa_s);
}());
//]]>
</script>
<!-- End Open Web Analytics Code -->
```

Figure 5 - Screenshot of REQAnalytics - JavaScript tracking code

The most valuable information which is collected with the help of this tool is related to the user clicks and user sessions. Every time a user clicks on a page, attributes of the click are saved in the database. The main attributes which are used by the recommender system in the analysis of the requirements are related to webpage URL, target URL of the click, DOM element, time and session.

4.1.4 Dashboard Visualization Report

This report, as presented in Figure 6, shows mostly information about the mapping status of requirements: number of (functional) requirements, number of mappings established, charts presenting the mapping status, list of requirements and their priorities and mapping status.

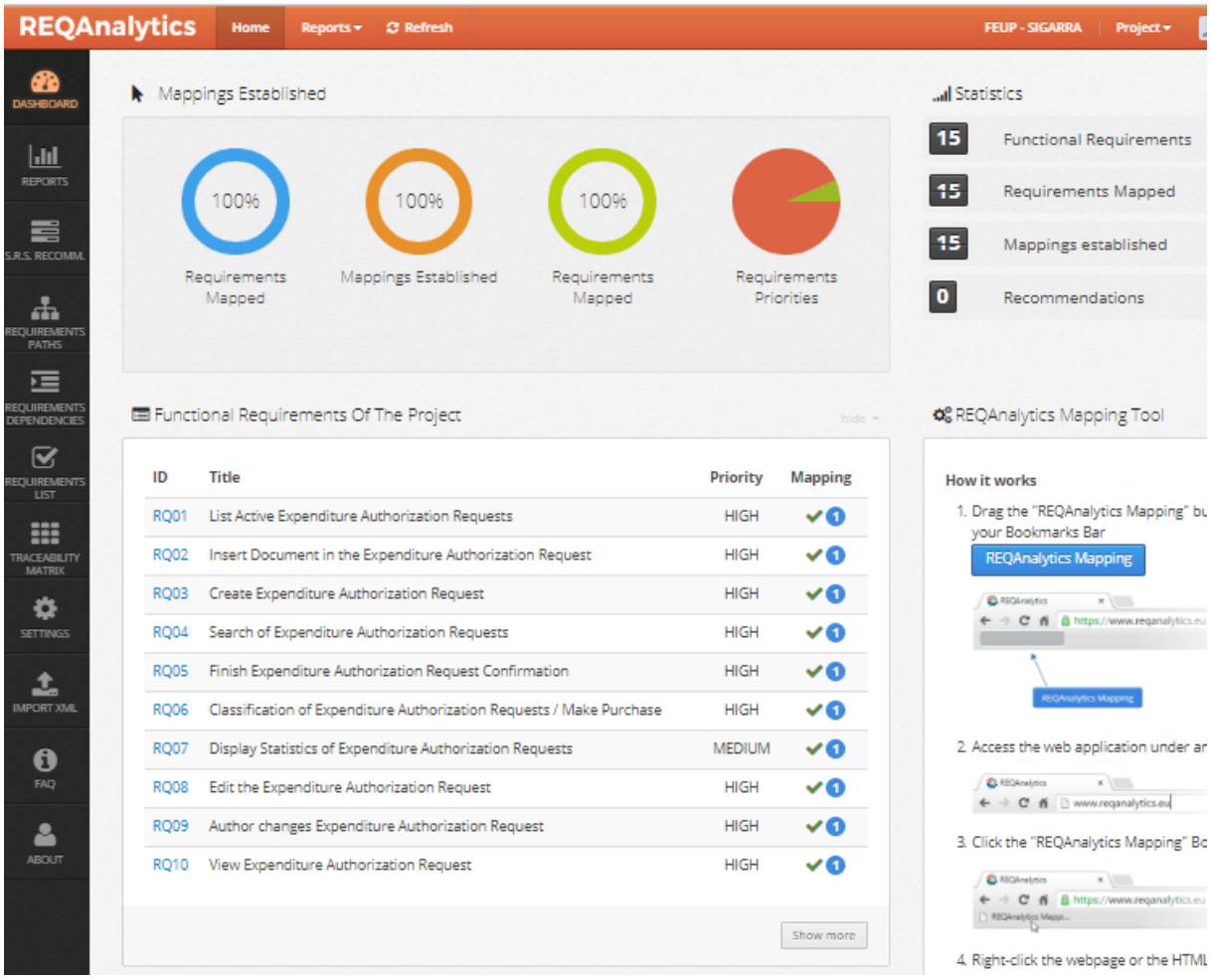

Figure 6 - Screenshot of REQAnalytics - Dashboard Visualization Report

4.1.5 Requirements Analytics Report

This feature comes with four types of reports: statistics about the number of clicks on each requirement, entry requirements, exit requirements and bounce requirements.

Entry requirements correspond to the pages which were first visited in a session and were mapped on requirements.

Exit requirements correspond to the pages which were last visited in a session and were mapped on requirements.

Bounce requirements correspond to the single-page sessions, which means that the session has the same start and end page. In this case, the user entered and left the website, without continuing to navigate through it, or seeing other pages.

4.1.6 Software Requirements Specification Recommendations

Based on the number of clicks on webpages or DOM elements mapped to a requirement, the system generates the following recommendations:

- Change Requirements Priority – where new priorities are automatically computed based on the popularity of a requirement – the more clicks on a requirement, the higher the priority;
- Create New Requirements – the system recommends mapping new requirements on unmapped web pages, but which are very popular among the users;
- Delete requirements – the system recommends deleting the requirements which have not been accessed in a given period;

4.1.7 Requirements Paths Report

By analysing the URLs accessed in a session, this feature computes and displays the most popular requirements paths through which the users navigate in their sessions, the most frequent requirements path patterns (workflows) and different metrics regarding reaching a goal.

4.1.8 Requirements Dependencies Network and Recommendations

The system generates a network which shows the relationships (dependencies) between requirements, recommends defining new dependencies between requirements based on the user navigation behaviour, and recommends splitting requirements in two or more requirements based on the number of clicks on the requirements.

4.1.9 Requirements List Report and Requirements Details Report

This list shows all the requirements, with more details than the list displayed in the dashboard. More information about a requirement is displayed in the requirement's details page. From this page, the requirements engineer can update the information about requirements, while having a broader view of the requirement, as well as to observe the requirement's web usage activity (number of visits, most visited page comparison, DOM element number of clicks, Top 5 DOM Elements clicked on the page)

4.1.10 Traceability Matrix Report

Most of existing techniques of traceability are used commonly between requirements and software test cases (Hayes, Dekhtyar & Osborne, cited in Garcia 2016), without being able to automatically generate and maintain traceability relationships (Garcia 2016).This Report proves that all the requirements were mapped on the webpages or DOM elements of the website. Also, this matrix helps to easily identify the requirements which were implemented in a certain webpage, and to provide a better visualisation of the mappings.

4.2 REQAnalytics - extensions

The following sections describe the extensions developed as part of this research work, with the purpose of answering the research questions. In addition to the extensions, some features of the system were updated such that they can be used for big sets of web usage data.

4.2.1 Frequent Path Patterns Report

This extension proposes a new feature of the REQAnalytics recommender system, in which the most frequent requirements path patterns (subsequences), having a user-defined length, are extracted and displayed for further analysis by the requirements engineer.

In every user session, the user navigates through the webpages in a certain order. As the functional requirements are mapped on URLs, this user navigation is used to determine the requirements session paths, which represent the order of accessing of requirements in the application.

In this feature, the user navigation through a website is analysed and the frequent subsequences of requirements (path patterns) are identified. For the requirements engineer, these patterns provide a better understanding of the user behaviour and navigation patterns in the context of requirements.

The requirements paths sequences are ordered by the number of their appearance in the user sessions. This way, the engineer can identify that some paths, which were not thought of as being popular, may appear to be quite the opposite.

Also, the patterns may help in understanding why the users quit a certain process, where the bottlenecks are, why a goal was not reached, which requirements sequences should be maintained with priority.

The path pattern discovery algorithm (APPENDIX A: Requirements sequence pattern discovery algorithm) starts by going through each requirements navigation path that has at least 2 steps. For each path, all the possible requirements sequences which it contains are identified.

These sequences will have a steps length between the minimum and maximum number of steps introduced by the engineer.

Figure 7 exemplifies the sequences extracted from 2 different navigation paths, having a length between 3 and 4 steps.

Figure 7 - Extracting patterns from paths

The algorithm then verifies if the combination exists in the other requirements paths, and whenever it is encountered, it is saved it in the $**match** array.

The **$patterns** array contains all the unique combinations and is created such that the patterns are not counted more times.

As an example, if the *RQ01RQ02RQ03* pattern is found in the first session (path), then this combination is saved in **$patterns** array and, by going through all the paths, every occurrence is saved in the **$match** array.

When moving to the second path, if the same *RQ01RQ02RQ03* pattern is identified in the path, then this pattern will not be searched for again in the paths.

Figure 8 exemplifies the non-overlapping appearances of a pattern inside the same navigation path.

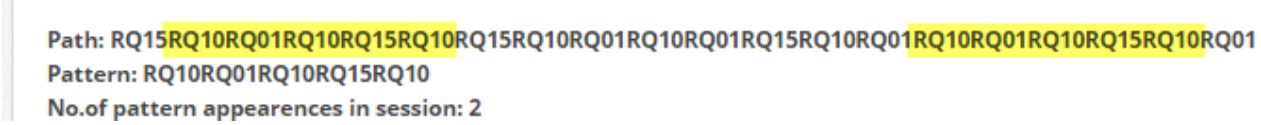

Figure 8 - Patterns appearances inside a navigation path

At the end, the algorithm groups the identical patterns from $**match** and counts how many times each pattern appears in all the sessions. The system then displays the patterns identified in a user-friendly report, as presented in Figure 9.

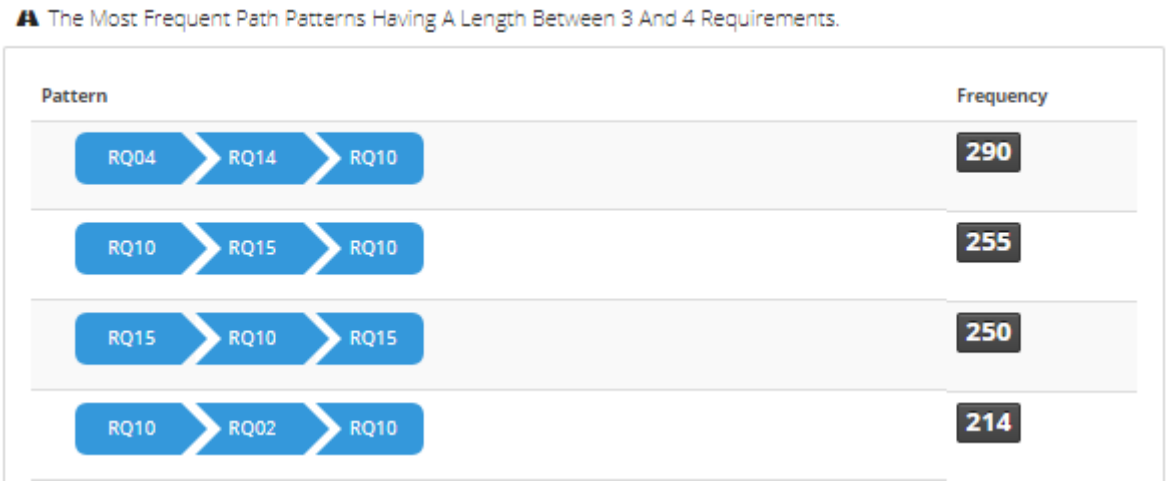

Figure 9 - Most frequent sequences report

4.2.2 Goals reports

This extension proposes a new feature of the REQAnalytics recommender system, in which the engineer selects the requirement that is associated to a goal. If a requirement is associated to a goal, then whenever a user accesses the requirement in his navigation, it is considered that the goal was reached.

The goals must be aligned with the business objectives, like, for example, increasing the number of subscribers or the number of purchases from the website.

After selecting a goal, the reports generated by this feature show information about the number of sessions where this goal was achieved, number of steps taken before reaching the goal and their frequency, the shortest sessions taken to reach the goal, the longest number of steps taken in a session to reach the goal, as well as the most frequent patterns that contain the goal.

These reports provide valuable information for the requirements engineer regarding the achievements of a goal from a web user perspective. The engineer can understand better what the users do before reaching the goal and what is their behaviour after reaching the goal.

Besides the requirements dependencies which can be shown in the frequent patterns, the engineer can identify the most frequent subsequences of requirements which lead to the goal achievement and make sure that these subsequences are maintained and improved such that the number of goal achievements increases.

The algorithm (APPENDIX B: Goals feature algorithm) starts by identifying the sessions (end-to-end) paths that contain the goal. It then computes the position where the requirement first appears in the path to identify the shortest and the highest number of steps needed to reach the goal, from the start of the session.

As exemplified in Figure 10, the goal can appear first in the same position in more than two different requirement paths.

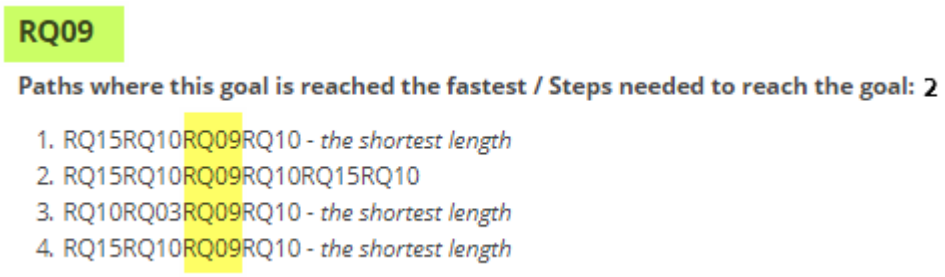

Figure 10 - Goal first reached at the same step in more navigation paths

The report displays the shortest navigation sessions where the goal was reached the fastest and the longest session where the user navigated through the highest number of steps until reaching the goal

The algorithm then computes the number of steps taken before reaching the goal, and their frequency, ordered by step number. This report, as exemplified in Figure 11, provides insight about how many steps the users usually take from the beginning of their navigation session until reaching the goal, and can help the engineer in identifying the necessity of reviewing the requirement if there are too many sessions where the goal is not reached fast.

Steps taken before reaching the goal	Sessions
0	2
1	17
2 *	36
3	10
4	11
5	6
6	7
7	6

Figure 11 - Steps taken before reaching a goal report

The algorithm continues by identifying the top 4 most frequent number of steps and extracting the minimum and maximum number from these steps.

It continues with identifying the subsequence patterns that contain the goal and have a length between the minimum and the maximum values extracted. This report, containing the subsequence patterns, shows the most common behaviour of the users before and/or after reaching a goal and, therefore, suggesting requirements subsequences which need to be maintained as they are the most preferred by the users in reaching the goal.

4.2.3 Software Requirements Specification Recommendations

This feature was updated to display correct recommendations when analysing big sets of data. The recommendations are now computed based on the average number of clicks on mapped requirements, such that the requirements engineer does not have to manually introduce a threshold, which is difficult to estimate in the context of analysing a new service, where prior traffic data is not available.

In this update, for each requirement, the number of clicks corresponding to the requirement is compared to this threshold, and to the minimum and maximum number of requirements clicks identified. The requirements are displayed from the most clicked to the least clicked.

If a requirement is having a total of accesses closer to the second most accessed requirement's number of accesses, but had a medium priority, then the system recommends increasing its priority to high. In the same way, if a requirement was labelled as having a medium priority, but the number of accesses is very low, closer to the lowest number of accesses, then the recommendation is to decrease the priority of the requirement.

In case the requirement's priority is high, but it has a very low number of accesses, then the new recommendation is to decrease it to medium, as the requirement, although not frequently accessed, may be very important for the well-functioning of the business process, since it was first labelled as of high priority.

In addition to this, the feature was adjusted to correctly calculate the number of clicks on each requirement. For each mapped URL, the system now sums the number of clicks on all its derivative URLs. Figure 12 better illustrates how the requirements are associated to derivative URLs of their mapped URL and that each of this URL can have multiple clicks (from different sessions and different users).

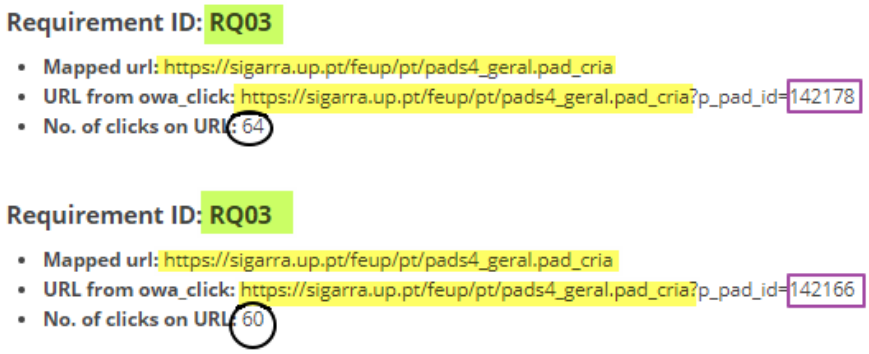

Figure 12 - Identifying number of clicks on a requirement

The recommendations for creating new requirements were also updated. For similar unmapped links, the feature now displays the suggested pattern URL to be mapped on the new requirement, as well as the suggested priority, based on the total number of clicks on these similar links.

Figure 13 shows the similar unmapped URLs which were discovered for a given unmapped URL, and the pattern which is generated by the *get_requirement_pattern()* function of the newly developed algorithm (APPENDIX B: Detect Common Patterns Algorithm) for creating new requirements.

Figure 13 - Detecting common patterns in URLs

4.2.4 System patches

In addition to the features presented above, different updates were applied to the system, as a response of its underperformance when analysing large sets of data. The updates were applied to the Reports, Most Used Navigation Paths and Requirements Dependencies features.

The main change in the application code was creating an array with all the clicks' URLs from the project and parsing the array, instead of querying the database to find each click URL. Wherever possible, the code was optimized to decrease the loading time of the system's recommendations and reports.

4.3 Discussions

This chapter starts by summarizing the problems identified in the Chapter 1.2 and presents the practical objectives of this research work. By implementing these practical objectives and validating them through the Case Study, the research questions will be answered.

The first section of this chapter presented the main workflow of the REQAnalytics recommender system, which was used in the Case Study from Chapter 5, as well as its key features. The following two sections of this chapter presented the new REQAnalytics features Frequent Path Pattern Report and Goals Reports, which were developed as part of this research work with the purpose to be used in the Case Study and answer the research questions.

The last two sections present the updates which were brought to the recommender system, in the Software Requirements Specification, Reports, Most Used Navigation Paths and Requirements Dependencies features, as a response to the system's underperformance when analysing the large volumes of data collected in the Case Study, and as well as a response to the identified recommendation improvement opportunities. The updates also help in the future research which is to be carried on this subject, involving the REQAnalytics recommender system.

5 Case study

This chapter goes into further detail about the case study steps, presented in the previous chapter.

The Case Study aims to answer the research questions from chapter 1.3, and consequently, to validate the extensions developed to the REQAnalytics recommender system and evaluate its ability to support the process of requirements management.

5.1 F64 Studio

This section presents the results of analysing the F64 (screenshot of homepage in Figure 14) web usage data, using the REQAnalytics recommender system.

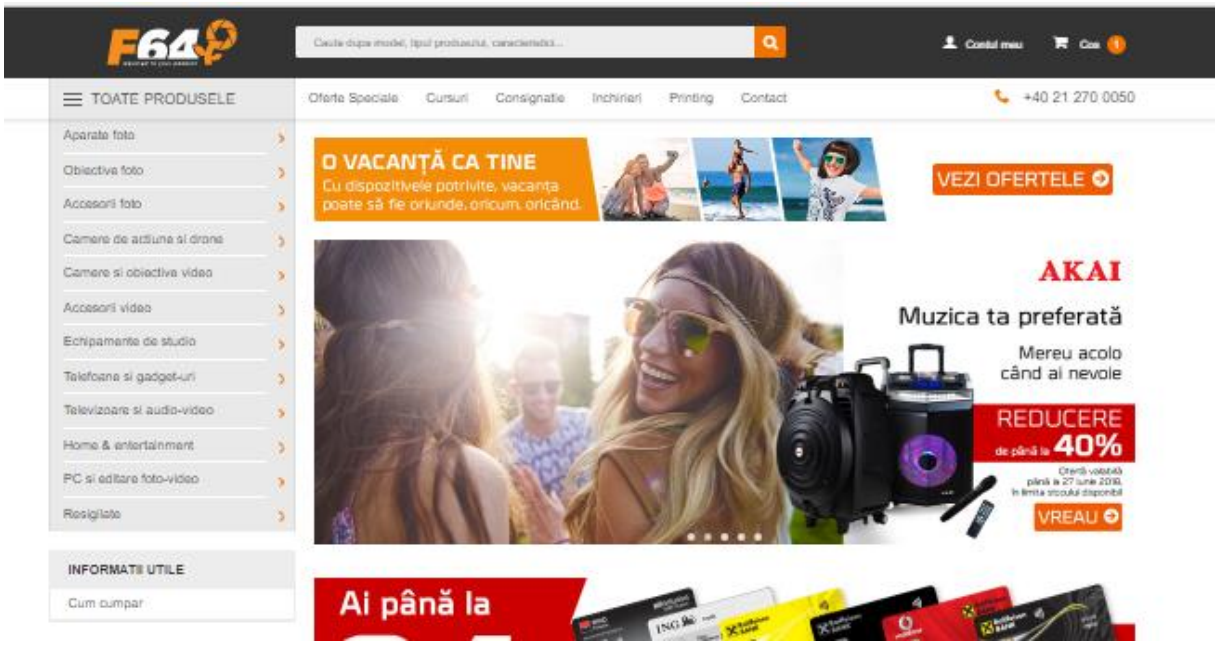

Figure 14 - Screenshot of F64 website - Home page

F64.ro is an e-Commerce Service, created in 2001, that has the following main functionalities:

- selling new electronic equipment and related accessories (for example, photography and video-related equipment and accessories, equipment for studios, phones and gadgets, TVs, PCs);
- selling consigned photography and video-related equipment and accessories (DSLR (digital single-lens reflex), SLR (single-lens reflex) and mirrorless cameras, camera lenses, video cameras and accessories);
- selling photography lessons and workshops (enrolment tax);
- providing information about equipment rental and printing services, which are available at the F64 headquarters;

As this is frequently accessed website, it is necessary to properly manage its functional requirements in order to maintain, but also to improve the quality of services delivered to the clients.

5.2 Goals

The following goals were defined for this case study:

- Analyse the recommendations automatically generated by the REQAnalytics system;
- Obtain most frequent subsequences of performed functionalities;
- Obtain the most frequent subsequences that contain a given goal;
- Obtain the most frequent subsequence's length performed before reaching a goal.

5.3 Data collection

To analyse the data, the first step that needs to be taken is to identify the functional requirements which were implemented in the website, and which will be used in the analysis performed by the REQAnalytics recommender system.

Since all the items and group items have unique URL, not following a certain pattern, as they are generated automatically based on the item or group name, it is not possible to map requirements on these URLs.

For example, the requirement "The website shall have a page to detail the item" cannot be mapped as these pages URLs are unique for every item, as exemplified in the Figure 15 below.

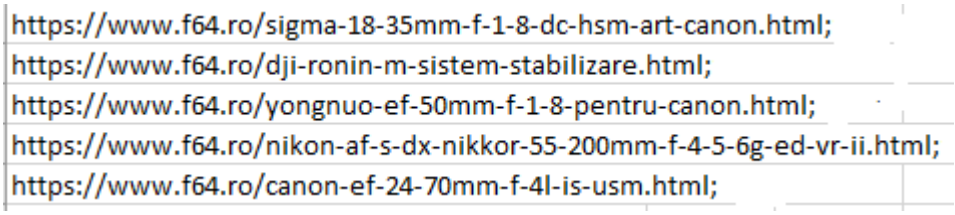

Figure 15 - Example of URLs for web pages containing item details

However, the URLs of the most relevant categories of items (cameras, camera lenses, photography accessories, action cameras and drones, video cameras and lenses, video accessories, resealed products) will be mapped on functional requirements, with the purpose of providing a better visualisation of the user navigation.

Given the fact that there are many requirements implemented in the service, for this Case Study, only a set of relevant functional requirements is selected.

Table 3 contains these functional requirements. For each functional requirement, the following attributes are presented: unique identifier (ID), title, description, priority.

Table 3 – F64 Functional requirements mapped in REQAnalytics

ID	Requirement title	Description	Priority
PH01	Homepage	The website shall have a homepage	High
PH02	Login page	The website shall have a login page	High

PH03	Sign-up page	The website shall have a sign-up page	High
PH04	Search field	The website shall provide a search field to search for products	High
PH05	Add item to shopping cart	The website shall have a button for checkout page (shopping cart)	High
PH06	Wishlist	The website shall have a Wishlist page	Medium
PH07	Complete order	The website shall have a checkout page for completing the order	High
PH08	Successful purchase	The website shall notify the customer if the purchase was successful	High
PH09	Delivery types	The website shall have a section with general information about types of delivery of the products	Medium
PH10	How to place order	The website shall have a section with general information about how to place an order	High
PH11	Contact page	The website shall have a Contact page	High
PH12	Renting services	The website shall have a section called Renting	Medium
PH13	Printing services	The website shall have a section with information about the printing services provided by the company	Low
PH14	Trainings and workshops	The website shall have a section with information about the trainings and workshops provided by the company	High
PH15	Consignment	The website shall have a section called Consignment	Medium
PH16	Special offers	The website shall have a section called Special offers	High
PH17	Cameras	The website shall have a section called Cameras which displays the types of cameras that are commercialized	High
PH18	Camera Lenses	The website shall have a section called Camera Lenses which displays the types of camera lenses that are commercialized	High
PH19	Photography Accessories	The website shall have a section called Photography Accessories which displays the types of photography-related accessories that are commercialized	High

PH20	Action Cameras and Drones	The website shall have a section called Action Cameras and Drones which displays the types of action cameras and drones that are commercialized	High
PH21	Video Cameras and Lenses	The website shall have a section called Video Cameras and Lenses which displays the types of video cameras and lenses that are commercialized	High
PH22	Video Accessories	The website shall have a section called Video Accessories which displays the types of video accessories that are commercialized	High
PH23	Resealed Products	The website shall have a section called Resealed which displays the types of products which are resealed and commercialized	High

The functional requirements described in Table 3 were exported to an XML document which was then imported in the REQAnalytics database.

After the requirements have been successfully imported, they are mapped with the web pages or URLs and DOM elements that uniquely identify every requirement. This mapping shows which are the exact implementations of the requirements in the website and is performed by the mapping tool of REQAnalytics recommender system.

The user interaction with the e-Commerce service was collected using the Open Web Analytics tool and stored in a database hosted by FEUP. After one month (26.03-26.04.2018) of web usage data collection, the tool had registered 246.488 distinct visitors, which requested 290.269 unique URLs or pages, and performed 3.827.520 clicks, over the course of 528.925 sessions.

Out of these, 240.539 sessions were bounce sessions, meaning that the users entered and immediately exited the website, without making any request for another page. The number of bounce sessions indicate the service's effectiveness in retaining visitors and encouraging them to navigate through the service.

In this Case Study only web usage data collected in 26.03-31.03.2018 is analysed, corresponding to 105.080 sessions and 746.738 clicks.

5.4 Analysis and results

Having the data collected in the database, and the requirements mapped in the system, it is now possible to generate reports and recommendations for the F64 website requirements.

5.4.1 Dashboard

The dashboard, as showed in Figure 16, provides information about the mapping status of the requirements. The report is easily interpreted by the requirements engineer and provides insight about requirements which have not been implemented in the website or which cannot be mapped on a page or DOM element.

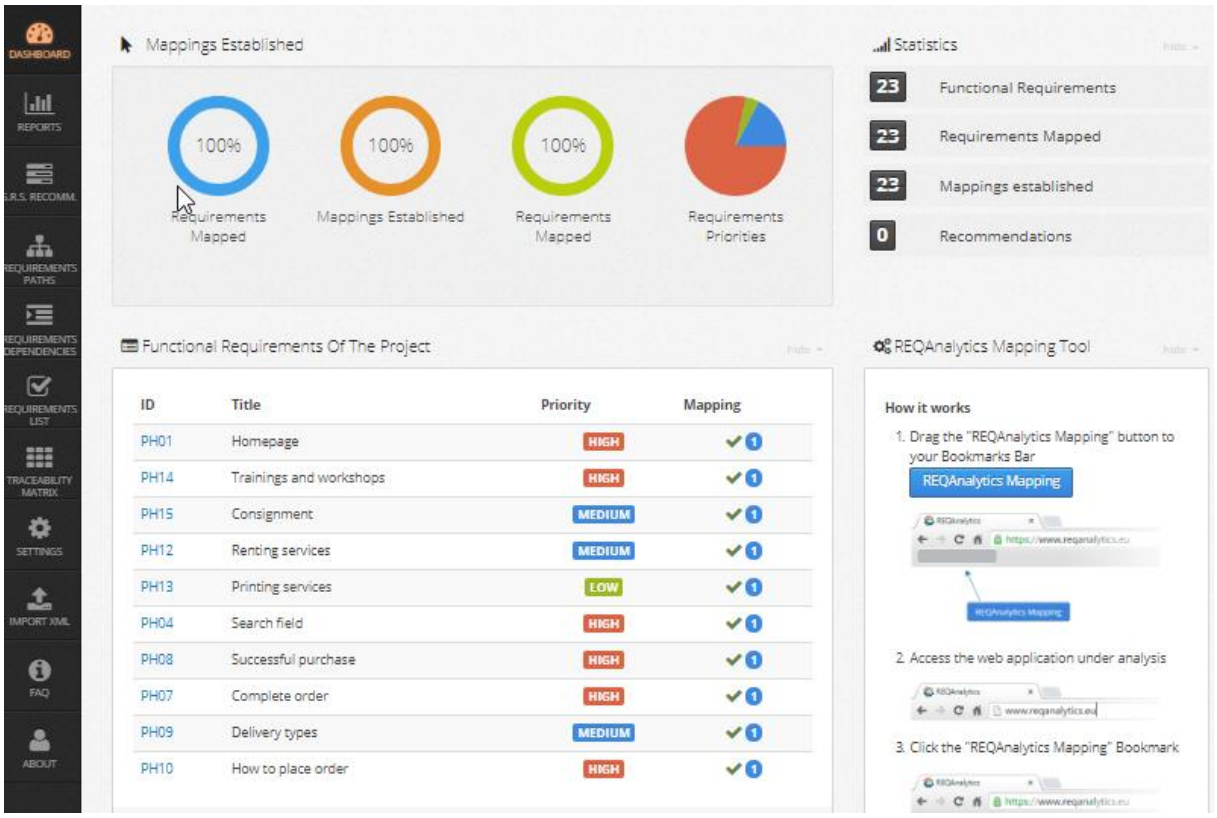

Figure 16 - F64 - Dashboard visualization report

For example, if there was a functional requirement that stated that a user which is a Gold client can unlock hidden offers in the "Special offers" page, then this functionality is implemented at the application level, with the hidden offers, being sent to the client level, and displayed in the same page and elements as the regular offers.

Therefore, if the requirements cannot be directly mapped on the page or element, their well-functioning should be revised at the application level, in the programming code, such that the information which is displayed or processed, is correct and complete.

5.4.2 Reports

The reports generated in this module provide information about the entry, exit and bounce requirements. As the requirements are mapped on pages, the entry requirements correspond to the pages which were first visited in a session and were mapped on requirements.

Figure 17 shows that the requirement PH01, corresponding to the homepage, was the first accessed requirement in a total of 29169 sessions.

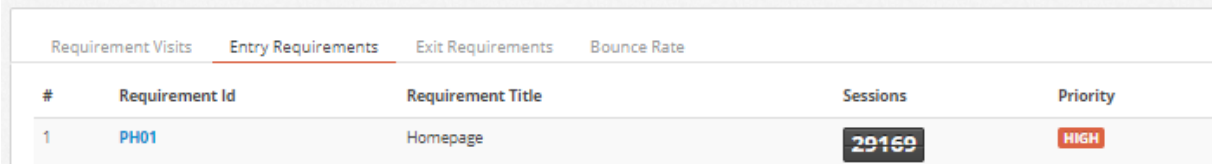

Figure 17 - F64 - Entry requirements

The exit requirements correspond to the pages which were last visited in a session and were mapped on requirements.

Figure 18 shows that the requirement PH01 corresponding to the homepage was the last accessed requirement in the highest number of sessions, but also that the visitors leave the service after adding items in the shopping cart or when they are required to access log in.

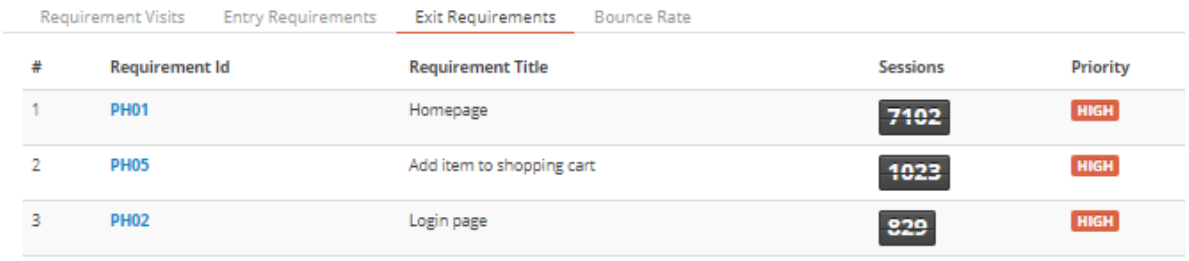

Figure 18 - F64 - Exit Requirements

These reports indicate that the homepage should be designed such that the most important functionalities of the service can be accessed directly from the homepage, as it is the most frequent entry point in the website. This also indicates that the requirement should be very well maintained as it is the main access point for the service.

Also, the requirements for adding items in shopping cart (PH05) and log in (PH02) should be revised to encourage the user to make the purchase or log in the website. Also, the PH02 requirement in the top exit requirements, may indicate that the users are not encouraged to create an account, so the requirements engineer may revise PH02 to attract the users to log in and continue the navigation through the service.

The bounce requirements correspond to the single-page sessions, which means that the session has the same start and end page. In this case, the user entered and left the website, without continuing to navigate through it, or seeing other pages.

For the F64 service, it is very important to encourage the users to navigate through the website, as some of the goals of the company is to increase the number orders and, implicitly, the number of items which are purchased.

Figure 19 shows that the PH01 requirement is a bounce requirement and, the webpage it corresponds to does not retain the user, meaning it does not encourage him to visit other pages. This requirement can be analysed by the requirements engineer and improvement opportunities can be identified, to reduce the number of bounce sessions which contain this requirement.

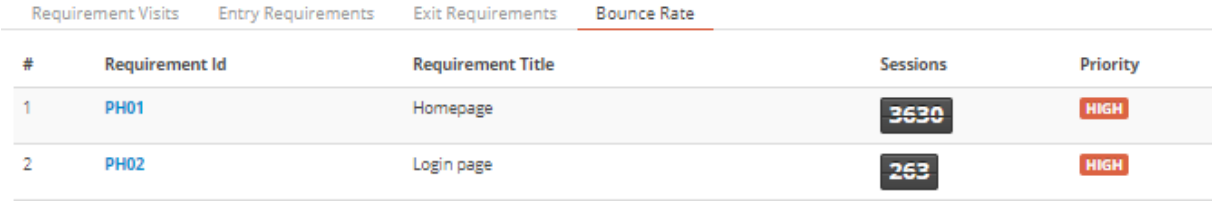

Figure 19 - F64 - Bounce requirements

5.4.3 Software requirements specification

- **Requirements Priority Change**

The recommendations to increase, decrease or maintain priorities are generated automatically based on the number of clicks in the pages which correspond to the requirements.

The recommendations help to identify whether the initial requirements priorities should be kept or changed, in accordance to users' preferences and navigation behaviour. A recommendation of "increase to high" proves that a requirement was underestimated in terms of frequency of access, and therefore be should be maintained with a high priority.

Figure 20 below shows the first 9 automatic recommendations for changing the requirements' priorities, which are generated by the system in the software requirements specification module.

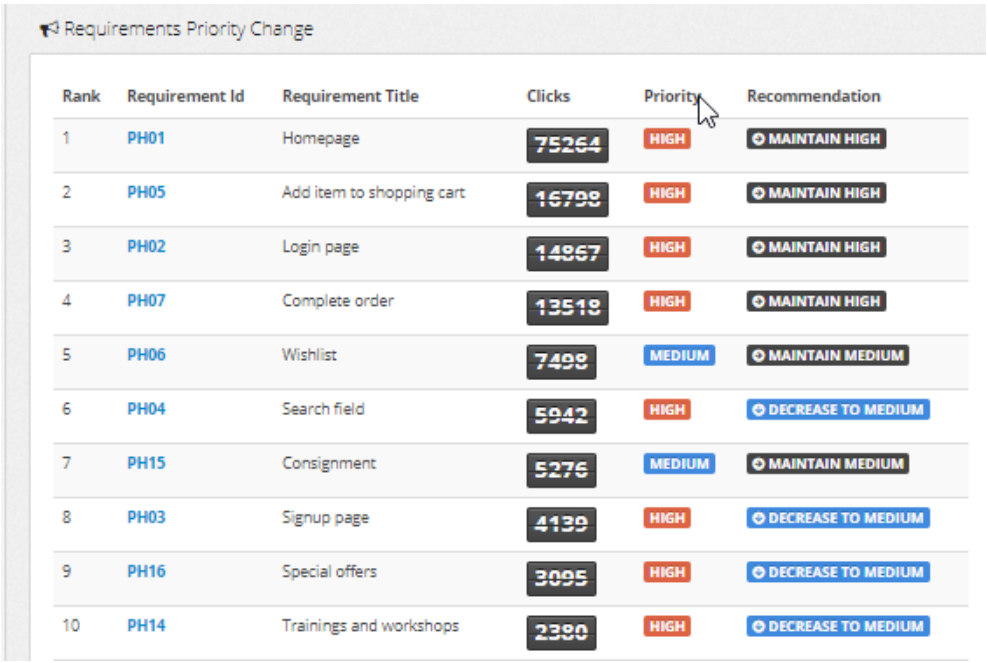

Figure 20 - F64 - Requirements Priority Change

These results show that users do not value the "Trainings and workshops", "Sign-up" and "Special offers" sections of the service as much as initially foreseen.

It can be observed that the popularity of the "Wishlist" and "Consignment" functionalities was estimated correctly as the users have less activity in these webpages than in login page or complete order page

- **Create new requirements**

Based on the information collected for the Requirements Priority Change report, the system generated the recommendations for creating new requirements, as seen in Figure 21.

These recommendations assist the requirements engineer in creating new requirements or mapping existing requirements in the system, by providing the URLs, number of similar links, number of clicks and suggested priority.

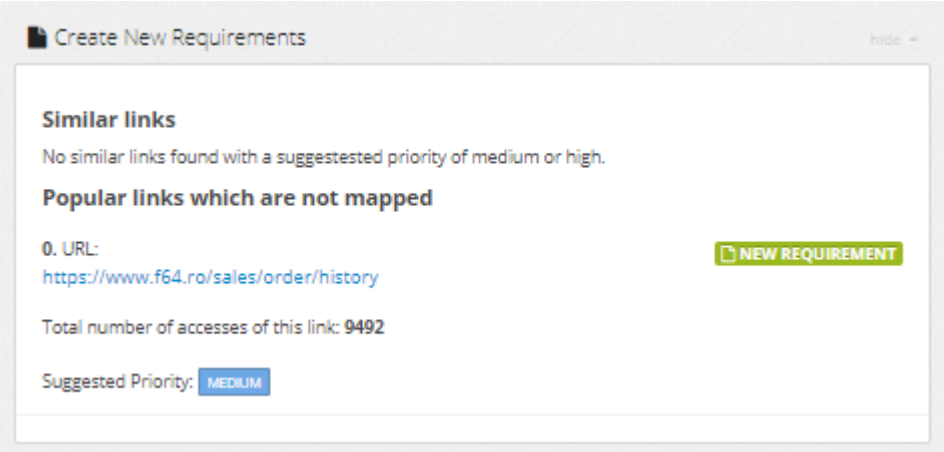

Figure 21 - F64 - Create new requirements

The recommendation generated is to create a new requirement for viewing the history of the orders, as the URL of this webpage was not mapped, but it presents a lot of user activity. This means that the users are interested in having access to this webpage, so by maintaining its newly created corresponding requirement, the service continues to satisfy the users' needs to have access to their order history.

As mentioned before, not all the requirements can be mapped on pages, URLs or elements. Similarly, not all pages, URLs or elements can be directly associated with a requirement, either because they are generated dynamically by the website's code (usually a framework), or because they were implemented to improve website usability and user experience (mostly the case of HTML elements).

In the F64 case, for each product which is commercialized on the website, there is a separate webpage, which is generated dynamically, at the application layer. This is a single functionality (The website shall have a details page for each item) that, at this moment, cannot be mapped in the recommender system.

However, even though information regarding the most purchased or favourite items can be extracted from other types of systems, by mapping a new requirement to the item's page, in the case the priority is high, is beneficial to making sure that the item's page is well maintained and is functioning properly.

For example, a requirement is mapped on a very-frequently purchased item's detail webpage. If, sometime after creating this requirement, the recommender system displays a recommendation to decrease the priority from high to medium, then it means that the item dropped in popularity. This may have various reasons, like the entry of new competitors on the market who sell this product, discounts on this item on other competitor's websites, or simply the fact that there was a new upgraded version released of this item.

However, the popularity may drop also because the information displayed in the webpage is not correct, accurate, or missing, which may happen for a series of reasons, like database records of the items being corrupted, accidentally deleted or updated, or data not being processed correctly by the application functions.

The measures taken to remediate the problems may affect other functionalities, therefore, having a good understanding of this webpage in terms of requirements traceability and dependability is valuable for the web developer, as well as the requirements engineer.

- **Delete requirements**

After the recommendations for creating new requirements, the system generates the suggestions to delete requirements which were not accessed in the period for which the analysis is made. In the F64, there is no suggestion to delete any requirements, as shown in Figure 22.

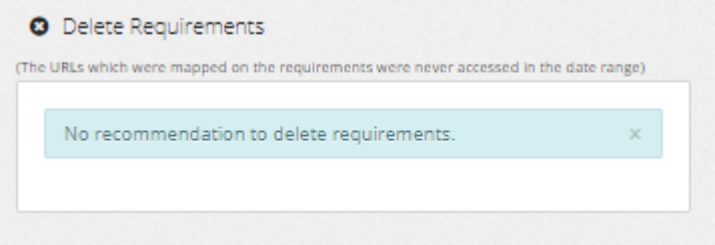

Figure 22 - F64 - Delete requirements

Nonetheless, this recommendation is useful to identify the requirements which, although implemented, were not used in a given period, as it provides a better insight about the requirement.

One advantage that can be obtained by removing unused requirements, and implicitly, their implementation, is easing the requirements maintenance process by focusing only on successfully deployed and used requirements. Another advantage is that updating their dependant requirements' implementations is becoming easier in terms of code complexity and resources involved.

In addition to the advantages presented above, by analysing the recommendations of changing the software requirements priorities, creating new requirements and deleting existing requirements, the software requirements specification is improved and kept up-to-date.

5.4.4 Navigation paths

- **Most Used Requirement Session Paths**

This report gives an insight on how the users navigate through the requirements and which are the most popular navigation paths, according to the number of these session paths (sessions where the users access the requirements in the same order, from the beginning to the end of the session).

The percentage from total number of the project's sessions can be seen as a navigation path rate. The higher the rate, the more focus should be placed on the navigation path.

Figure 23 exemplifies the 5 most used requirements paths in the F64 project.

This report shows that the most popular path that the users navigate through is the one in which the access the home page (PH01) and start searching for products items (PH04). This exact navigation path is encountered in more than 8% of the total user sessions from the period of analysis. This is an indicator that the PH04 requirement (Search field) should be maintained with priority and maybe improvements of the Search field can be identified.

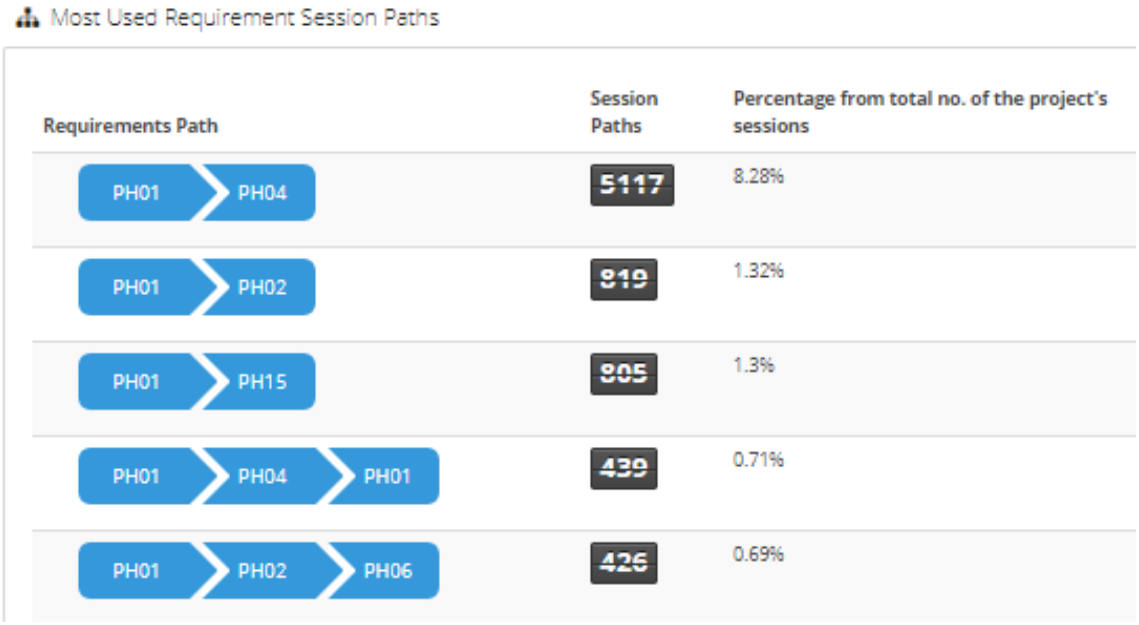

Figure 23 - F64 - Most used requirements paths

The second most popular requirement path is the one in which the user accesses the web home page (PH01), logs in (PH02) and then leaves the service. Some of the users, however, choose to verify their wish list (PH06) before exiting the session. This exact navigation path is encountered in more than 8% of the total user sessions from the period of analysis. The percentage can be seen as a navigation path rate. The higher the rate, the more focus should be placed on the navigation path.

- **Most frequent subsequences**

These subsequences can be started at any point in a web session. By analysing them, the requirements engineer gets a better understanding of the user behaviour and the navigation patterns and may identify how the service can be improved in terms of user navigation. Figure 24 shows the 5 most frequent requirements subsequences in the F64 project, which have a length between 3 and 4 steps (the length is defined by the REQAnalytics user).

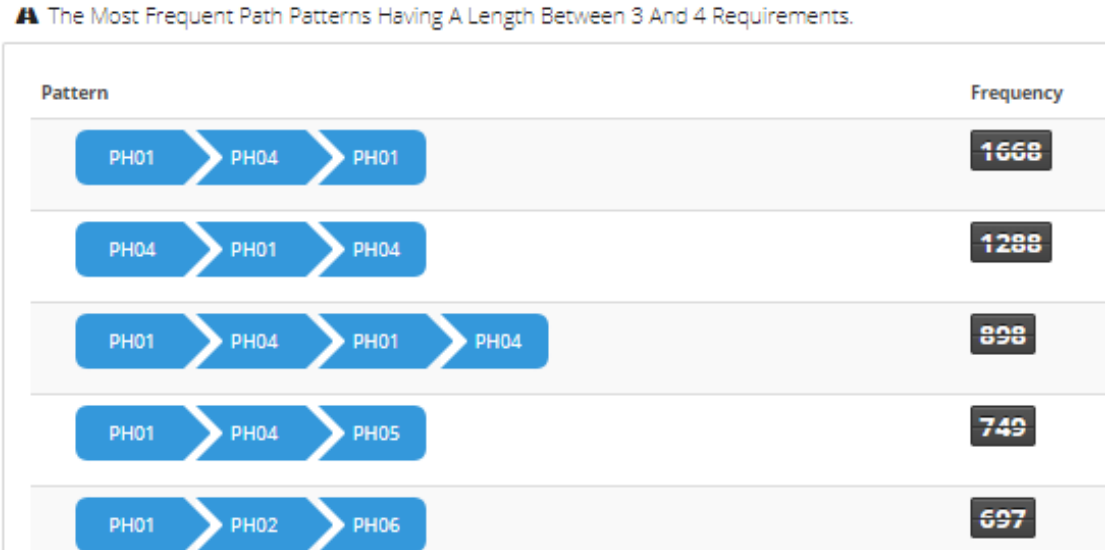

Figure 24 - F64 - Most frequent requirements subsequences

The requirements paths subsequences are ordered by the number of their appearance in the user sessions. This way, the engineer can identify that some paths, which were not thought of as being popular, may appear to be quite the opposite. Also, the patterns may help in understanding why the users quit a certain process, where the bottlenecks are, why a goal was not reached, or which requirements subsequences should be maintained with priority.

In this report, the most common subsequence of patterns consists in accessing the home page (PH01), searching for products (PH04) and then accessing the home page again (PH01). This shows, similarly as for the most used navigation paths, that the PH04 requirement (Search field) should be maintained with priority and maybe improvements of the Search field can be identified. In the context of e-Commerce services, this behaviour is normal and does not present any element of surprise. Still, this "searching" experience should provide the customers with the best and relevant results, so they can be encouraged to add items in the products, and, ultimately, purchase them.

However, one interesting subsequence that appears to be very popular, according to this report is the one in which the user accesses the home page (PH01), logs in (PH02) and checks his Wishlist (PH06). This shows that the users value this Wishlist functionality and the requirements engineer may identify opportunities of improving it, by making the Wishlist functionality more visible for the users.

Figure 25 below presents a proposal of improvement of the service, based on the analysis made on the Most Frequent Path Patterns report. The proposal is to add a shortcut to the Wishlist page, in the user account administration page. The user account administration page is the is the first page that a user sees after he logs in.

The "Don't forget your favorites" section randomly displays products from the user's Wishlist and, therefore, users are reminded about their favorite items at every log in. This improvement has the purpose of promoting this feature and constantly reminding the users about the products they liked, hoping that their next step is to buy them.

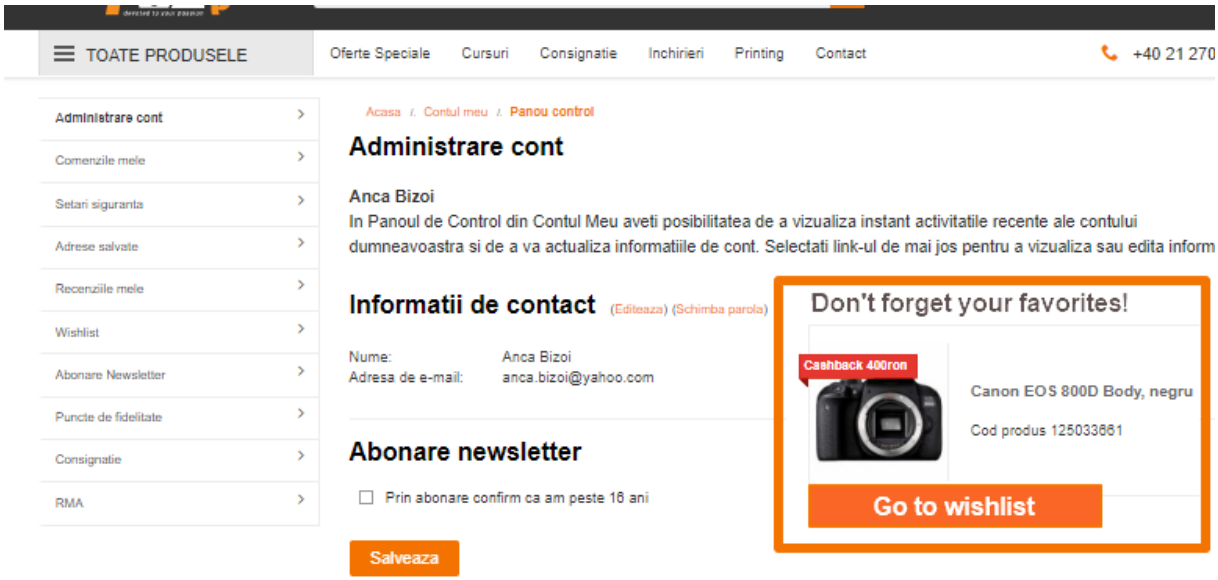

Figure 25 - F64 - Improvement of User Account Administration Page

- **Goals**

The goals must be aligned with the business objectives, like, for example, increasing the number of subscribers or the number of purchases from the website.

After selecting the requirement which corresponds to a goal, the requirements engineer can analyse the reports generated and get different information about the goal achievements.

As the web usage data was collected from an e-Commerce service, the reports which are detailed below, were generated after selecting requirement PH08 (The website shall notify the customer if the purchase was successful), which corresponds to the business goal of increasing the number of online purchase orders.

- **Steps taken before reaching the goal**

As showed in Figure 26, the report shows that users reach this goal mostly after two steps in their requirements path. However, this goal is reached very often from the third or fourth steps.

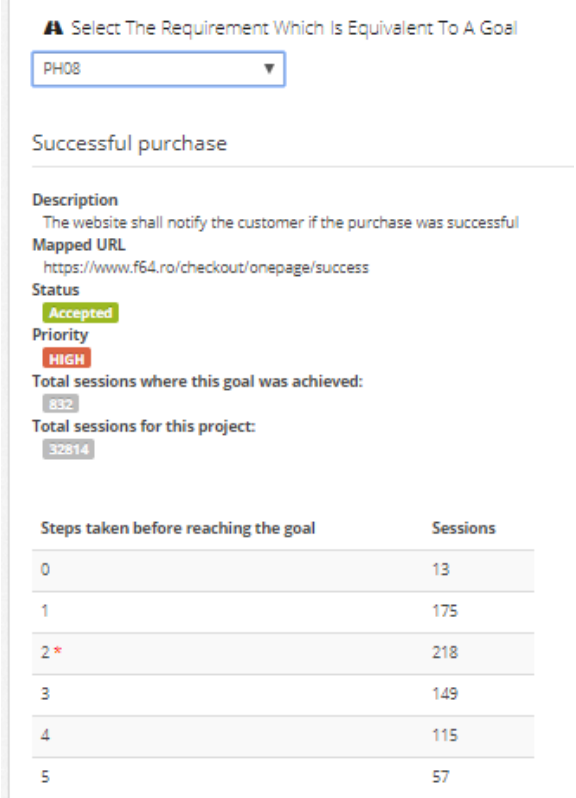

Figure 26 - F64 - Goals report

These values are normal for this e-Commerce service, as the clients first must log in, and then complete the order details and payment, for the purchase to be successful. Nonetheless, the number of steps taken before reaching a goal is a good indicator of how fast a goal is reached in a user navigation path.

The higher the number of steps taken before reaching a goal, the higher the chances of the user to quit the process or to not be satisfied with the service provided.

This service is configured to display in the same web page (corresponding to the PH07 requirement), all the 5 sections necessary to complete the purchase: client billing information, delivery options, delivery method, payment method and order review.

Therefore, the navigation of the user throughout these sections is not recorded to be furtherly analysed if the users find it difficult to reach the button of submitting the order.

However, the frequency of small number of steps necessary to complete the order, although normal given the number of requirements mapped in the system, raises some questions since it would suggest that the customers are very well-trained users and do not have questions about the delivery methods (showed in the webpage corresponding to PH09) or about how to place the order (showed in the webpage corresponding to PH10).

To better understand the users' behaviour in reaching the goal of successfully submitting an order, the other reports are analysed.

- **Shortest sessions taken to reach the goal**

By analysing the first report presented in Figure 27, corroborated with the information obtained from the first report presented in Figure 26 it results that very few navigation paths contain just this goal.

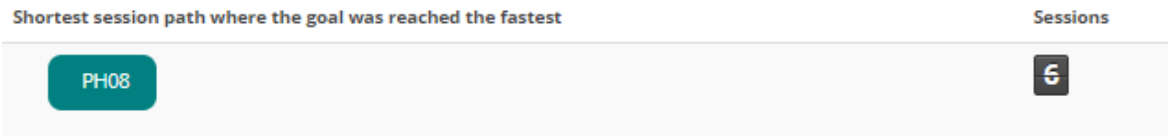

Figure 27 - F64 - Shortest session path where the goal was reached the fastest

One reason would be that a user closed the browser immediately after the page of successfully submitted order was displayed, and then the user reopened the browser later and the last page viewed was reloaded. If this is the case, the application code should be updated such that the home page is displayed, instead of the successful order page (as this would confuse the user and not encourage his navigation through the service).

Another reason would be that the users access unmapped pages before reaching this goal. In this case, those pages should be identified and mapped on the functional requirements, for the system to generate more accurate reports.

- **The longest number of steps taken in a session to reach the goal**

By analysing the second report presented in Figure 27, it results that the requirements PH09 and PH10 were not accessed not even in the longest session.

In addition to this, it shows to the requirements engineer, a more detailed visualization about how users navigate through the service.

In this case, the user accesses the service, searches for products, probably browsing the search results items (as their links are not mapped), adding products to shopping cart, accessing the page to submit order, changing his mind, continuing to browse and search, until finally submits the order, after which continues to navigate through the service.

The requirements engineer, understands that there is no unusual user behaviour in this navigation path.

Figure 28 - F64 - The longest number of steps taken in a session to reach the goal

- **The most frequent patterns that contain the goal**

Figure 29 presents the 5 most frequent subsequences which contain the goal associated to the PH08 requirement.

According to this report, the most common last steps that the users take before reaching the goal are PH05 (Add item to shopping cart) and PH07 (Complete order – shopping cart checkout).

This finding is relevant, as the engineer must understand why the system reports that after adding an item to cart, the order appears to be successfully submitted, although PH07 is the last artefact the users click on before reaching the goal.

The engineers can identify that there is another (unmapped) web page from where the users can submit their order. If so, a new requirement should be mapped on that web page.

Figure 29 - F64 - Most frequent sequences that contain a goal

According to this report, there is no frequent pattern where the users accessed PH09 (Delivery types) or PH10 (How to place order) before reaching the goal.

As a result of the analysis of these reports, the web page that implements PH07 (Complete order) was analysed and it was identified that the web page's footer displays the links for the web pages corresponding to PH09 and PH10, as shown in Figure 30.

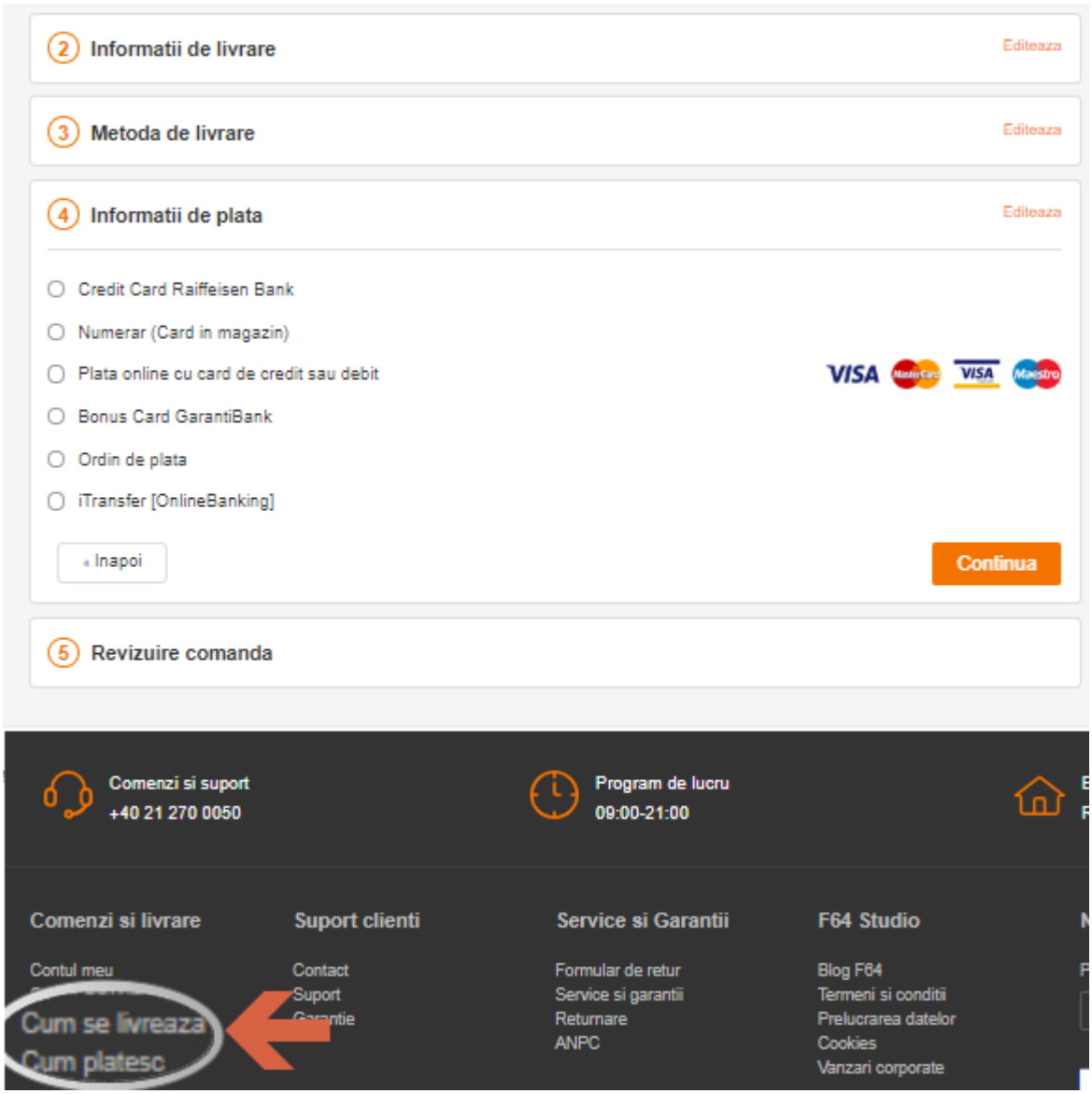

Figure 30 - F64 - Screenshot of Complete order page

Therefore, if the users, while completing the forms needed to place the order, have doubts or questions about the order or delivery, they may open those links in new browser tabs. By doing so, using the right click of the mouse, for example, the system does not record the action as being a step in the user's navigation path.

This information is valuable for the engineer, as he understands that the updates of the web page's footer may affect the user navigation paths, and he must consider this relationship between the checkout sections and checkout web page footer.

By displaying links for PH09 and PH10 inside the checkout sections, it is ensured that the user navigation is not affected by any changes in the web page's footer.

Requirements dependencies

- **Create New Requirements Dependencies**

Figure 31 shows one of the recommendations to create new requirements dependencies, as generated by the system. By analysing the new requirements dependencies recommendations, the requirements engineer can manage the change requests better. In this case, it is recommended to create a dependency between the "Sign-up" page and the "Successful purchase". Therefore, if a change request is submitted for PH03 (Sign-up), the requirements engineer should analyse the impact this kind of change would have on the "Successful purchase" page (PH08).

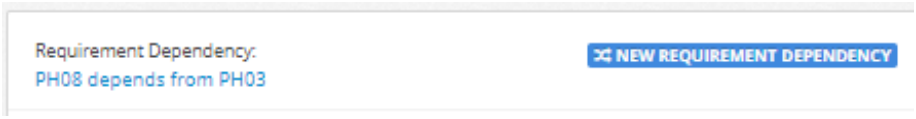

Figure 31 - F64 - Requirements dependencies network

- **Split Requirement**

This recommendation is displayed when there is a substantial activity in a page that is mapped on the requirement. Figure 32 exemplifies three automatically generated recommendations to split a requirement in two or more requirements.

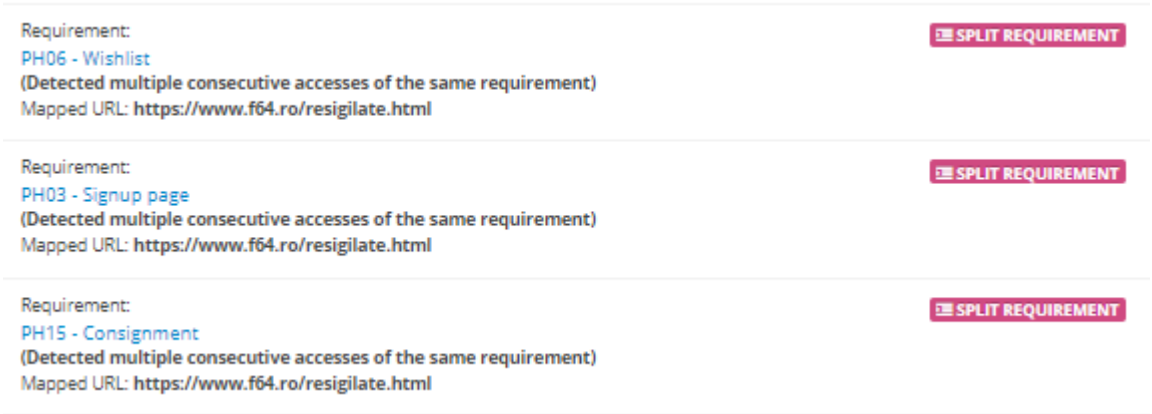

Figure 32 - F64 - Split requirements recommendation

In this case, for example, the sign-up page requirement can be split in more functional requirements, each for every important section of the sign-up form. The requirements can be formulated like: "The Sign-up page shall have a form where the customers can enter their data", "The sign-up form shall have a "Remind me" option" and "The sign-up page form shall display the "Terms and Conditions of Use"".

Traceability Matrix

As shown in Figure 33, the traceability matrix proves that all the requirements were mapped on URLs. Also, this matrix helps to easily identify the requirements which were implemented in a certain webpage, and to provide a better visualisation of the mappings.

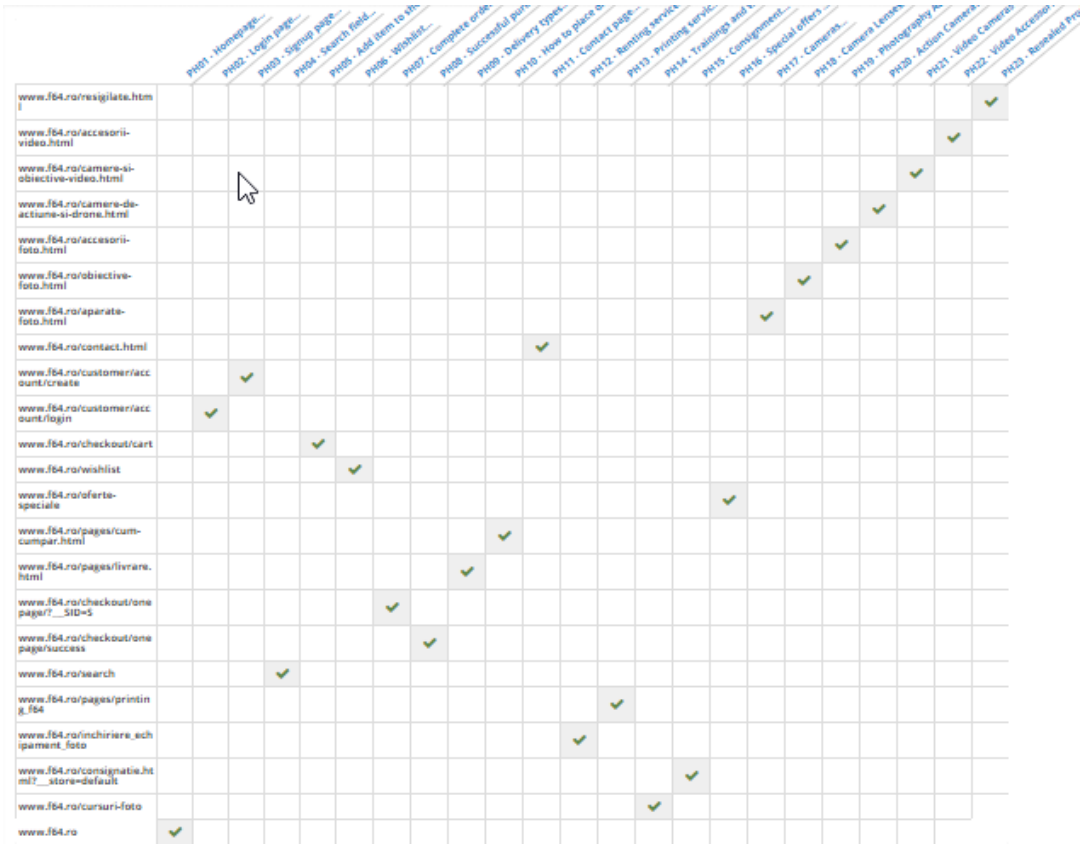

Figure 33 - F64 - Traceability Matrix

5.5 Discussion

All the goals that were proposed in Chapter 5.2 were achieved in this Case Study. Web usage data collected from the F64.ro e-Commerce service was used in the analysis performed using the REQAnalytics recommender system.

With the achieved results, the research questions formulated in the first Chapter are answered, as follows:

RQ1. Is it possible to use web usage mining for requirements management activities, with the help of REQAnalytics recommender system?

> Yes. The reports and recommendations generated by the REQAnalytics recommender system, based on the web usage data, were analysed and useful information that can be used in the requirements management activities was extracted.
>
> The Entry / Exit / Bounce Requirements reports help in identifying the requirements that can be revised to increase the service's capacity of retaining users and encouraging them to navigate through the service.
>
> The Requirements Priority Change recommendations help to identify whether the initial requirements priorities should be kept or changed, in accordance to users' preferences and navigation behaviour.

The Create New Requirements recommendations assist the requirements engineer in creating new requirements or mapping existing requirements in the system, by providing the URLs, number of similar links, number of clicks and suggested priority.

The Delete Requirements recommendations are useful to identify the requirements which, although implemented, were not used in a given period.

The Most Used Requirement Session Paths report gives an insight on how the users navigate through the requirements and which are the most popular navigation paths, according to the number of these session paths

The Create New Requirements Dependencies recommendation assist the requirements engineer in creating new requirements dependencies or update existing ones, for a better management of change requests.

The Split Requirement recommendations show the functionalities which present a significative amount of user click activity, for the requirements engineer to further analyse the requirements and identify the requirements which can be formed by splitting the requirement.

The Traceability Matrix makes it easier for the requirements engineer to identify the artefacts that implement the requirements.

RQ2: Are the most frequent subsequences of performed functionalities, generated with the REQAnalytics recommender system, useful for the requirements management activities?

Yes. By analysing them, the requirements engineer gets a better understanding of the user behaviour and the navigation patterns and may identify how the service can be improved in terms of user navigation.

RQ3: Are the most frequent subsequences that contain a given goal, generated with the REQAnalytics recommender system, useful for the requirements management activities?

Yes. By analysing them, the requirements engineer gets a better understanding of the user behaviour and the navigation patterns that contain a given goal, and he may identify how the service can be improved in terms of user navigation

RQ4: Is the most frequent subsequence's length performed before reaching a goal, generated with the REQAnalytics recommender system, useful for the requirements management activities?

Yes. The number of steps taken before reaching a goal is a good indicator of how fast a goal is reached in a user navigation path. An unusual value of this number, corroborated with the analysis of different other goals-based reports, may lead the requirements engineer into finding relevant information about the users' navigation behaviour before reaching a goal.

6 Conclusion and future research

6.1 Conclusion

As presented by Garcia and Paiva (2016f), analyzing the usage of websites can help identify improvements and help to maintain the website and its software requirements. To validate this idea, the REQAnalytics recommender system was developed. This system generates recommendations and reports about a service's functional requirements, based on the web usage data that was collected by a web analytics tool (OWA).

Given the challenges that still exist in the requirements management activities, corroborated with the potential of improvement in the existing functionalities, this research work proposes an evolution of the REQAnalytics tool, by extending the analysis performed over the sequences of functionalities (requirements) and refining the data provided for Software Requirements Specification. This evolution has the purpose of helping the requirements engineers in the requirement maintenance activities, and to improve the overall quality of the services.

In addition to this, the following existing features were updated: Reports, Most Used Navigation Paths and Requirements Dependencies. The updates were necessary for analysing the large volumes of data collected in the Case Study, and as well as a response to the identified recommendation improvement opportunities. The updates also help in the future research which is to be carried on this subject, involving the REQAnalytics recommender system. In this research work, a Case Study was used to validate the proposed extensions.

A summary of how the research work results responded to the practical objectives is presented in the following paragraphs.

To identify if it is possible to use web usage mining for requirements management activities, the recommendations and reports generated by the REQAnalytics recommender system, including the ones developed for this research work, were analysed. Besides the features which will be described next, the following reports and recommendations were analysed: Dashboard Visualization Report, Requirements Analytics report (entry / exit / bounce requirements report), Most Used Navigation Paths, Requirements Dependencies Network and Recommendations and the Traceability Matrix Report.

To obtain the most frequent subsequences of performed functionalities, the Frequent Path Pattern Report feature was developed. This report presents the most frequent subsequences of performed functionalities, having a user-defined length.

To identify the most frequent subsequences that contain a given goal and to calculate the most frequent subsequence's length performed before reaching a goal, the Goals Reports feature was developed. In this feature, if a requirement is associated to a goal, then whenever a user accesses the requirement in his navigation, it is considered that the goal was reached. The reports generated by this feature show information about the number of sessions where this goal was achieved, number of steps taken before reaching the goal and their frequency, the shortest sessions taken to reach the goal, the longest number of steps taken in a session to reach the goal, as well as the most frequent patterns that contain the goal.

To identify if it is possible to improve the recommendations to create new requirements, a refining of the data provided for Software Requirements Specification Recommendations was performed. This feature was updated to display correct recommendations of changing requirements' priorities. The recommendations for creating new requirements were also

updated. For similar unmapped links, the feature now displays the suggested pattern URL to be mapped on the new requirement, as well as the suggested priority, based on the total number of clicks on these similar links.

6.2 Future research

While developing the extensions of REQAnalytics, as well as analysing the data that was collected, different improvement opportunities were identified.

As mentioned in the previous chapter, not all the requirements can be mapped on pages, URLs or elements. Similarly, not all pages, URLs or DOM elements can be directly associated with a requirement, either because they are generated dynamically by the website's code (application layer), or because they were implemented to improve website usability and user experience (mostly the case of HTML elements). Therefore, one improvement of the system would be to have the capability of mapping a requirement on multiple different URLs, which do not follow a certain pattern, as they may be dynamically generated based on their content.

Another feature which may improve the REQAnalytics ability to support the requirements management activities, is analysing the evolution of the requirements in a given date range, by comparing different parameters (number of requirements accesses, number of entry/ exit/ bounce requirements, DOM elements clicks in requirements).

The evolution may help predicting the dates or time periods when the requirements are very accessed by the users, and, therefore, they should be marked in the system as having a "peak" period. This way, the requirements engineer can better estimate the optimal time periods for service updates, new deployments, or implementation of change requests, as changing the service in a period of high-usage may lead to service failures and user dissatisfaction.

Although the REQAnalytics system generates recommendations for splitting requirements, it would be interesting to provide more details in these recommendations, that could help the requirements manager in identifying the new functionalities. One approach in implementing this feature is for the system to analyse the DOM elements which were clicked in a website and recommend creating new functionalities which should be mapped on the most clicked DOM elements.

One more feature that may help especially in the requirements traceability activity using the REQAnalytics system, is grouping the functional requirements in service components or workflows. One functional requirement can be found in more service components or workflows and the latter can be expressed as a subsequence of requirements. This feature may provide a better visualization for the requirements engineer of the business processes (workflows) implemented in the service. At the same time, he can have a better overview of the relationships between service components (features), when changes in business processes appear or changes in components are requested. Also, this feature can show different classifications of workflows and service components, based on user navigation behaviour.

Other improvement that can increase the accuracy of the recommendations generated by REQAnalytics is adding an extra attribute to the requirements which are mapped, which is "risk". This attribute would make the requirements change priority recommendation more accurate, as a requirement should continue having a high priority even if it is not clicked very often. Not managing properly this kind of requirements may lead to service failures or the service incapacity to deliver what expected. For example, if a service has a feature of generating monthly reports, then the requirements mapped for this feature will not be very accessed,

suggesting a low priority. However, failure in generating those reports may seriously affect the business functions, hence, it is important to mark the requirement as having a high risk.

Until this point, only the web data provided by the user navigation behaviour has been used in the process of requirements management. However, analysing the information about the web user may help deliver new recommendations or reports. For example, by analysing the visitor's country, as collected by the web analytics tool, it may be identified that certain requirements are very accessed from a certain country so an improvement of the requirement by adding page translation can be recommended.

To sum up, REQAnalytics has been proven to be effective in assisting the requirements management process, but there are still improvements which can be brought to the recommender system, as presented above, and which may be the subject of future research work.

References

Internet World Stats, 2017, *World Internet Users and 2018 Population Stats*. Available from: http://www.internetworldstats.com/stats.htm [15 May 2018].

Netcraft, 2018, *January 2018 Web Server Survey*. Available from https://news.netcraft.com/archives/2018/01/19/january-2018-web-server-survey.html [15 May 2018].

Garcia, JE & Paiva, ACR 2016a, "Maintaining Requirements Using Web Usage Data", *Conference on ENTERprise Information Systems / International Conference on Project MANagement / Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN / HCist*, Porto, Portugal, 5 - 7 Oct. 2016, vol.100, pp. 626 - 633, 2016, https://doi.org/10.1016/j.procs.2016.09.204.

Standish Group, 1995, "CHAOS". Available from: https://www.projectsmart.co.uk/white-papers/chaos-report.pdf [15 May 2018].

Wei, GT, Kho, S, Husain, W & Zainol, Z 2015, "A Study of Customer Behaviour Through Web Mining", *Journal of Information Sciences and Computing Technologies*, ISSN 2394- 9066, vol. 2, issue 1, pp.103 107. Available from: http://www.scitecresearch.com/journals/index.php/jisct/article/view/40/20 [20 May 2018].

Garcia, JE & Paiva, ACR 2016b, "A Requirements-to-Implementation Mapping Tool for Requirements Traceability", in Journal of Software, vol.11, no. 2, pp. 193 –200, https://doi.org/10.17706/jsw.11.2.193-200.

ISO/IEC/IEEE 2017, "12207-2017 - ISO/IEC/IEEE International Standard - Systems and software engineering - Software life cycle processes" ,https://doi.org/10.1109/IEEESTD.2017.810077.

IEEE 2014, "730-2014 - IEEE Standard for Software Quality Assurance Processes", https://doi.org/10.1109/IEEESTD.2014.6835311.

ISO/IEC 2014, "ISO/IEC 14143-2:2011 Information technology -- Software measurement -- Functional size measurement -- Part 2: Conformity evaluation of software size measurement methods to ISO/IEC 14143-1".

ISO/IEC 2017, "ISO/IEC 19770-1:2017 Information technology -- IT asset management -- Part 1: IT asset management systems—Requirements".

Hayes, JH, Li, W & Rahimi, M 2014, "Weka meets TraceLab: Toward convenient classification: Machine learning for requirements engineering problems: A position paper", *2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*, Karlskrona, Sweden, 26-26 Aug. 2014, pp. 9-12, https://doi.org/10.1109/AIRE.2014.6894850.

Sunner, D & Bajaj, HK 2016, "Classification of Functional and Non-functional Requirements in Agile by Cluster Neuro-Genetic Approach", *International Journal of Software Engineering and Its Applications*, vol. 10, no. 10, pp. 129-138, http://dx.doi.org/10.14257/ijseia.2016.10.10.13.

Ott, D 2013, "Automatic Requirement Categorization of Large Natural Language Specifications at Mercedes-Benz for Review Improvements". In: Doerr J., Opdahl A.L. (eds) *Requirements Engineering: Foundation for Software Quality*. REFSQ 2013. Lecture Notes in Computer Science, vol 7830, pp. 50-64. Springer, Berlin, Heidelberg, https://doi.org/10.1007/978-3-642-37422-7_4.

Cleland-Huang, J, Settimi, R, Zou, X & Solc, P 2007, "Automated classification of non-functional requirements", *Requirements Engineering*, vol. 12, no. 2, pp. 103–120, https://doi.org/10.1007/s00766-007-0045-1.

Eckhardt, J, Vogelsang, A & Fernández, DM 2016, "Are "non-functional" requirements really nonfunctional?", *38th International Conference on Software Engineering ICSE '16*, Austin, Texas, 14, 22 May 2016, pp. 832 842. https://doi.org/10.1145/2884781.2884788.

Winkler, J & Vogelsang, A 2016, "Automatic Classification of Requirements Based on Convolutional Neural Networks", *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*, Beijing, China, 12-16 Sept. 2016, pp. 39-45, https://doi.org/10.1109/REW.2016.021.

Dollmann, M & Geierhos, M 2016, "On- and Off-Topic Classification and Semantic Annotation of User-Generated Software Requirements" *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, 1-5 Nov. 2016, pages 1807–1816. Available from: http://www.aclweb.org/anthology/D16-1186 [18 May 2018].

Glinz, M 2014, "A Glossary of Requirements Engineering Terminology", Version 1.6, Available at: https://www.ireb.org/content/downloads/1-cpre-glossary/ireb_cpre_glossary_16_en.pdf [27 Apr. 2018].

Aoyama, M, Nakatani, T, Saito, S, Suzuki, M, Fujita, K, Nakazaki, H & Suzuki, R, 2010, "A Model and Architecture of REBOK (Requirements Engineering Body of Knowledge) and Its Evaluation", *2010 Asia Pacific Software Engineering Conference*, Sydney, Australia, 30 Nov.-3 Dec. 2010, pp. 50-59, https://doi.org/10.1109/APSEC.2010.16.

Pohl, K 2014, "Requirements Engineering: Fundamentals, Principles, and Techniques", Springer.

VersionOne 2018, "12th Annual State of Agile Survey", Available from: https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report/ [3 June 2018].

Heikkilä, VT, Damian, D, Lassenius, C & Paasivaara, M 2015, "A Mapping Study on Requirements Engineering in Agile Software Development," *2015 41st Euromicro Conference on Software Engineering and Advanced Applications*, Funchal, Portugal, 26-28 Aug. 2015, pp. 199-207, https://doi.org/10.1109/SEAA.2015.70.

Kassab, M 2014, "An Empirical Study on the Requirements Engineering Practices for Agile Software Development", *2014 40th Euromicro Conference Series on Software Engineering and Advanced Applications*, Verona, Italy, 27-29 Aug. 2014, pp. 254-261, https://doi.org/10.1109/SEAA.2014.77.

Inayat, I, Salim, SS, Marczak, S, Daneva, M & Shamshirband, S 2015, "A systematic literature review on agile requirements engineering practices and challenges", *Computers in Human Behavior*, vol. 51, part B, pp. 915-929, https://doi.org/10.1016/j.chb.2014.10.046.

Gaikwad, V & Joeg, P 2017, "A case study in requirements engineering in context of agile", *International Journal of Applied Engineering Research*, vol. 12, no. 8, pp. 1697-1702. Available from: https://www.ripublication.com/ijaer17/ijaerv12n8_31.pdf [28 May].

Wiegers, KE 2003, "Software Requirements", third edition, Microsoft Press, Redmond, USA.

Jayatilleke, S & Lai, R 2018, "A systematic review of requirements change management", *Information and Software Technology*, vol. 93, pp. 163-185, https://doi.org/10.1016/j.infsof.2017.09.004.

Oliveros, A, Napolillo, F & Infesta, FL 2016, "Requirements in Web applications development", *IEEE CACIDI 2016 - IEEE Conference on Computer Sciences*, Buenos Aires, Argentina, 30 Nov.-2 Dec. 2016, pp. 1-5, https://doi.org/10.1109/CACIDI.2016.7786002.

McKinsey Global Institute 2016, “The age of analytics: competing in a data-driven world”. Available from: https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/the-age-of-analytics-competing-in-a-data-driven-world [20 May 2018].

IDC White Paper 2017, “Data Age 2025: The Evolution of Data to Life-Critical”. Available from: https://www.seagate.com/www-content/our-story/trends/files/Seagate-WP-DataAge2025-March-2017.pdf [20 May 2018].

Larose, D & Larose, C 2014, “Discovering Knowledge in Data: An Introduction to Data Mining”, Second edition, John Wiley & Sons, Inc., Hoboken, New Jersey.

Liu, B 2011, “Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data”, Second edition, Springer Heidelberg, Dordrecht.

Singh, C & Kautish, SK 2015, “Page Ranking Algorithms for Web Mining: A Review”, Available from: International Journal of Computer Applications (IJCA): https://research.ijcaonline.org/icaet2015/number5/icaet4066.pdf [18 May 2018].

Saini, S & Pandey, HM 2015, “Review on Web Content Mining Techniques”, *International Journal of Computer Applications*, vol. 118, no. 18, pp. 33-36, https://doi.org/10.5120/20848-3536.

Trappey, AJC, Trappey, CV, Chang, A-C & Chen, LWL 2016, “Using Web Mining and Perceptual Mapping to Support Customer-Oriented Product Positions and Designs”, Borsato, M. et al. (eds.) *Transdisciplinary Engineering: Crossing Boundaries*, IOS Press, pp. 533-542, https://doi.org/10.3233/978-1-61499-703-0-533.

Honest, N, Patel, B, & Patel, A 2015, “A Study of User Navigation Patterns for Web Usage Mining”, *International Journal of Advent Research in Computer and Electronics*, vol. 2, no. 1. Available from: https://www.researchgate.net/publication/275713001_A_Study_of_User_Navigation_Patterns_for_Web_Usage_Mining [18 May 2018].

Pokorný, J & Smizanský, J 2005, “Page content rank: an approach to the web content mining”, *International Conference on Applied Computing*, Algarve, Portugal, 22-25 Feb. 2005, pp. 289-296. Available from: https://www.researchgate.net/publication/220970122_Page_content_rank_an_approach_to_the_web_content_mining [18 May 2018].

Pushpalatha, N & Reddy, SSS 2017, "Towards an extensible web usage mining framework for actionable knowledge", *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, Coimbatore, India, 10-11 March 2017, pp. 35-40, https://doi.org/10.1109/ICICCT.2017.7975232.

Verma, N, Malhotra, D, Malhotra, M & Singh, J 2015, "E-commerce Website Ranking Using Semantic Web Mining and Neural Computing", *Procedia Computer Science*, vol. 45, pp. 42-51, https://doi.org/10.1016/j.procs.2015.03.080.

Raheja, N & Katiyar, VK 2014, "Efficient web data extraction using clustering approach in web usage mining", *International Journal of Computer Science Issues*, ISSN: 1694-0814, vol. 11, issue 1, no 2, pp. 216-225. Available from: https://pdfs.semanticscholar.org/e479/8673803bcf97c3124d96705c3c56aa4062d9.pdf [18 May 2018].

Cooley, R, Mobasher, B & Srivastava, J 1999, "Data Preparation for Mining World Wide Web Browsing Patterns", *Knowledge and Information Systems*, vol. 1, issue 1, pp. 5-32, https://doi.org/10.1007/BF03325089.

Bošnjak, S, Marić, M & Bošnjak, Z 2010, "The Role of Web Usage Mining in Web Applications Evaluation", *Management Information System*s, vol. 5, no. 1, pp. 31-36. Available from https://www.researchgate.net/publication/265243206_The_Role_of_Web_Usage_Mining_in_Web_Applications_Evaluation [18 May 2018].

Ambreen, T, Ikram, N, Usman, M & Niazi, M 2016, "Empirical research in requirements engineering: trends and opportunities", *Requirements Engineering*, vol. 23, no. 1, pp. 63-95, https://doi.org/10.1007/s00766-016-0258-2.

Spoletini, P & Ferrari, A 2017, "Requirements Elicitation: A Look at the Future Through the Lenses of the Past", *2017 IEEE 25th International Requirements Engineering Conference (RE)*, Lisbon, Portugal, 4-8 Sept. 2017, pp. 476-477, https://doi.org/10.1109/RE.2017.35.

Garcia, JE & Paiva, ACR 2016c, "Manage Software Requirements Specification Using Web Analytics Data", *World Conference on Information Systems and Technologies - Trends and Advances in Information Systems and Technologies*, Naples, Italy, 27-29 March 2018, vol. 2, pp. 257–266, https://doi.org/10.1007/978-3-319-77712-2_25.

Garcia, JE & Paiva, ACR 2016d, "An Automated Approach for Requirements Specification Maintenance", in Rocha, A, Correia, A, Adeli, H, Reis, L & Mendonca Teixeira, M. (eds), *New Advances in Information Systems and Technologies*, vol. 444, pp. 827-833, Springer, https://doi.org/10.1007/978-3-319-31232-3_78.

Garcia, JE & Paiva, ACR 2016e, "REQAnalytics: A Recommender System for Requirements Maintenance", *International Journal of Software Engineering and Its Applications*, vol. 10, no. 1, p. 129-140, https://doi.org/10.14257/ijseia.2016.10.1.13.

OECD 2015, “Frascati Manual 2015: Guidelines for Collecting and Reporting Data on Research and Experimental Development, The Measurement of Scientific, Technological and Innovation Activities”, OECD Publishing, Paris, pp 44-45, http://dx.doi.org/10.1787/9789264239012-en.

Creswell, JW 2003, “Research design: Qualitative, quantitative, and mixed method approaches”, Fourth edition, Sage Publications, Los Angeles.

Kowalczyk, D 2016. *Research methodologies: Quantitative, qualitative, and mixed methods [video file]*. Available from http://study.com/academy/lesson/research-methodologies-quantitative-qualitative-mixed-method.html [10 June 2018].

Denzin, NK & Lincoln, YS (eds.) 2000, “Handbook of qualitative research”, Second edition, Sage Publications, Thousand Oaks.

Creswell, JW & Creswell, JD 2017, “Research Design: Qualitative, Quantitative, and Mixed Methods Approaches”, Fifth edition, London, Sage Publications.

Creswell, JW 2003, “Research design: Qualitative, quantitative, and mixed method approaches”, Second edition, Sage Publications, Thousand Oaks.

Teddlie, C & Tashakkori, A 2009, “Foundations of mixed methods research”, Sage Publications, Los Angeles.

Collis, J & Hussey, R 2009, “Business Research: A practical guide for undergraduate and postgraduate students”, third edition, Palgrave Macmillan.

Yin, RK 1994, “Case study research: Design and methods”, Thousand Oaks, Sage Publications.

Jackson, RL, Drummond, DK & Camara, S 2007, "What Is Qualitative Research?", *Qualitative Research Reports in Communication*, vol. 8, no. 1, pp. 21- 28, https://doi.org/10.1080/17459430701617879.

Garcia, JE 2016, “Requirements Change Management based on Web Usage Mining”, PhD thesis, University of Porto. Available at: https://www.researchgate.net/publication/308887831_Requirements_Change_Management_based_on_Web_Usage_Mining.

Garcia, JE & Paiva, ACR 2016f, “An Automated Approach for Requirements Specification Maintenance”, *World Conference on Information Systems and Technologies - New Advances in Information Systems and Technologies*, Recife, Brazil, 22-24 March 2016, vol. 1, pp. 827–833, Available from: https://www.researchgate.net/publication/295402600_An_Automated_Approach_for_Requirements_Specification_Maintenance [19 May 2018].

Bibliography

Laplante, PA 2017, “Requirements Engineering for Software and Systems”, Third Edition, Auerbach Publications.

Bourque, P & Fairley RE, (eds.) 2014, “Guide to the Software Engineering Body of Knowledge, Version 3.0”, IEEE Computer Society.

Dick, J, Hull, E & Jackson, K 2017, “Requirements Engineering, Fourth edition, Springer International Publishing.

Multisoft Virtual Academy 2016, *BABOK V2 Vs. V3 – Comparing The Knowledge Areas,* viewed 28 April 2018, http://www.multisoftvirtualacademy.com/blog/babok-v2-vs-v3-comparing-the-knowledge-areas-2/.

Powoh, TV 2016, “Research Methods-Quantitative, Qualitative, and Mixed methods”, https://doi.org/10.13140/RG.2.1.1262.4886.

Victor, SP & Rex, MX 2016, “Analytical implementation of web structure mining using data analysis in educational domain”. Available from: https://www.researchgate.net/publication/300045793_Analytical_implementation_of_web_structure_mining_using_data_analysis_in_educational_domain [31 May 2018].

Irfan, S & Ghosh, S 2018, “Web Mining for Information Retrieval”, International Journal of Engineering Science and Computing, vol. 8, no. 4, pp. 17277 – 17283. Available from:

http://ijesc.org/upload/3b875c5ea1777fbd8836e3419447df18.Web%20Mining%20for%20Information%20Retrieval.pdf [03 June 2018].

APPENDIX A: Requirements subsequence pattern discovery algorithm

```php
<?php
    $min_steps = $_POST['min_steps'];
    $max_steps = $_POST['max_steps'];
    $req_list_final = $_SESSION['req_list_final'];
    $patterns = [];
    $match = [];
    $no_of_patterns_in_session = 0;
    $i = 0;
    foreach ($req_list_final as $key => $path) {
        $i = $min_steps;
        $path_temp = $path;
        while($i <= (strlen($path)/4) and $i <= $max_steps){
            while (strlen($path_temp) >= ($i*4)) {
                $reqs = substr($path_temp, 0, $i*4);
                if(!in_array($reqs, $patterns)){
                    $patterns[] = $reqs;
                    foreach ($req_list_final as $key1 => $session) {
                        if((strlen($session)/4) >= $min_steps){
                            if(substr_count($session, $reqs) > 0) {
                                $no_of_patterns_in_session = substr_count($session, $reqs);
                                for($k = 1; $k <= $no_of_patterns_in_session ; $k++)
                                {
                                    $match[] = $reqs;
                                }
                            }
                        }
                        else {
                            if($key != $key1){
                                unset($req_list_final[$key1]);
                            }
                        }
                    }
                }
                $path_temp = substr($path_temp, 4);
            }
            $path_temp = $path;
            $i++;
        }
        echo '</ol>';
    }
    $patterns_totals = array_count_values($match);
    arsort($patterns_totals);
?>
```

APPENDIX B: Goals feature algorithm

```php
<?php

    include "db.php";
    $selected = $_POST['selected'];
    $req_list = $_SESSION['req_list'];
    $mysqli = new mysqli($hostname, $user, $pass, $database);
    $sql = "SELECT requirement_title, status, description, priority, url
          FROM req_requirement , req_map
          WHERE req_requirement.requirement_id = req_map.requirement_id
          AND  req_requirement.requirement_id = '".$selected."'";

    $Result = $mysqli->query($sql);

    $row = $Result->fetch_object();
    $title = $row->requirement_title;
    $description = $row->description;
    $priority = $row->priority;
    $status = $row->status;
    $url = $row->url;

    $j=0;
    $paths_with_req = [];
    $positions = [];
    $total_paths = 0;

    $total_paths_with_goal = 0;
    foreach ($req_list as $key => $value) {
        if($value!=''){
            $pos = strpos($value, $selected);
            if($pos !== FALSE){
                $positions[] = $pos;
                $paths_with_req[] = $value;
                $total_paths_with_goal +=1;
            }
            $total_paths++;
        }
    }
    $totals = array_count_values($positions);
    ksort($totals);
    $totals_temp = $totals;
    $shortest_step = key($totals_temp)/4;
    $val = end($totals_temp);
    $longest_step = key($totals_temp)/4;
```

```
42      $longest_step = key($totals_temp)/4;
43
44      $shortest_paths = [];
45      $longest_paths = [];
46      $length_shortest_path = 0;
47      foreach ($paths_with_req as $value) {
48          $pos = strpos($value, $selected);
49          if($pos == (($shortest_step)*4)){
50              $shortest_paths[] = $value;
51
52              if($length_shortest_path == 0)
53                  $length_shortest_path = strlen($value);
54              else
55                  if($length_shortest_path > strlen($value) )
56                      $length_shortest_path = strlen($value);
57          }
58
59          if($pos == (($longest_step)*4)){
60              $longest_paths[] = $value;
61          }
62      }
63
64      foreach($shortest_paths as $key=>$value){
65          if(strlen($value) != $length_shortest_path){
66              unset($shortest_paths[$key]);
67          }
68      }
69
70      $shortest_paths_sorted = array_count_values($shortest_paths);
71      $longest_paths_sorted = array_count_values($longest_paths);
72
73      $req_list_final = $_SESSION['req_list_final'];
74      $patterns = [];
75      $match = [];
76      $totals_temp_patterns = $totals;
77      arsort($totals_temp_patterns);
78      $pattern_steps = [];
79      $min_steps = 0;
80      $max_steps = 0;
81      $i = 0;
```

APPENDIX C: Detect Common Patterns Algorithm

```php
<?php
    $totals = $unmapped_url;
    arsort($totals);
    $count = 0;
    $count1 = 0;
    $similar_url_list = [];
    $similar_url_list_similars = [];
    $urls_with_patterns = [];
    $count_similar_url_access = [];
    $recpattern = [];
    $no_similar_links =[];
    //function for the url pattern
    function get_requirement_pattern($urls)
        {
          //create an array with urls
            $urls = array_map('strtolower', array_map('trim', $urls));
            $sort_by_strlen = create_function('$url1, $url2', 'if (strlen($url1)
                == strlen($url2)) { return strcmp($url1, $url2); } return (
                strlen($url1) < strlen($url2)) ? -1 : 1;');
            usort($urls, $sort_by_strlen);
            $common_pattern = '';
            //split the shortest url in an array containing its characters
            $shortest_url = str_split(array_shift($urls));
                //go through every character from the shortest url
            foreach ($shortest_url as $ci => $char) {
                // the  pattern  is obtained by adding every character to the
                common_pattern string; check in every similar url if it contains
                the common_pattern
                $common_pattern.= $char;
                foreach ($urls as $ui => $url) {
                    if (!strstr($url, $common_pattern)) {
                        break 2; //not go through the other characters if the
                        pattern is no longer matched
                    }
                }
            }
            return substr($common_pattern, 0, -1);
        }
```

```php
// Detect Common Patterns Algorithm
// go through each unmapped url
foreach ( $totals as $key_one => $count_one ) {
    //for this url, introduce the number of his accesses in the count_similar_url_access array
    $count_similar_url_access[$key_one] = $count_one;
    //the url is saved in the $urls_with_patterns array
    $urls_with_patterns[] = $key_one;
        foreach ($totals as $key_two => $count_two ) {
            //verify if the url was already found to be similar with other url; if not, verify if it is similar to other url
            if(!in_array($key_one, $similar_url_list_similars))
            {
                //compute how similar the urls are
                similar_text(strtolower($key_two), strtolower($key_one), $percent);
                if ($percent > 90 and $percent < 100) {
                    if(!in_array($key_one, $similar_url_list) && !in_array($key_one, $similar_url_list_similars)){
                        $similar_url_list[] = $key_one;
                        $similar_url_list_similars[] = $key_two;
                        $count_similar_url_access[$key_one] += $count_two;
                        $urls_with_patterns[] = $key_two;
                    }
                    else{
                        $similar_url_list_similars[] = $key_two;
                        $count_similar_url_access[$key_one] += $count_two;
                        $urls_with_patterns[] = $key_two;
                    }
                }
            }
        }
        //after getting all the similars for this urls, verify which is their common pattern
        if(count($urls_with_patterns) != 1){
            //pattern array will contain the common pattern of the similar links
            $pattern[$key_one] = get_requirement_pattern($urls_with_patterns);
            $no_similar_links[$key_one] = count($urls_with_patterns);
        }
        $urls_with_patterns = [];
}
?>
```