

Abner Cleto Filho

**Linking User Stories to UX elements:  
Recommendations to Reduce the Virtual  
Navigational Distances**

**Sorocaba, SP**

**27 de Março de 2020**



Abner Cleto Filho

**Linking User Stories to UX elements:  
Recommendations to Reduce the Virtual Navigational  
Distances**

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação (PPGCC-So) da Universidade Federal de São Carlos como parte dos requisitos exigidos para a obtenção do título de Mestre em Ciência da Computação. Linha de pesquisa: Engenharia de Software e User eXperience.

Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

Programa de Pós-Graduação em Ciência da Computação – PPGCC-So

Supervisor: Prof. Dr. Luciana A. M. Zaina

Sorocaba, SP

27 de Março de 2020

---

Cleto Filho, Abner

Linking User Stories to UX elements:

Recommendations to Reduce the Virtual Navigational Distances/ Abner Cleto  
Filho. – 2020.

174 f. : 30 cm.

Dissertação (Mestrado) – Universidade Federal de São Carlos – UFSCar

Centro de Ciências em Gestão e Tecnologia – CCGT

Programa de Pós-Graduação em Ciência da Computação – PPGCC-So.

Supervisor: Prof. Dr. Luciana A. M. Zaina

Banca examinadora: Prof. Dr. Luciana A. M. Zaina, Profa. Dra. Sabrina dos Santos

Marczak, Prof. Dr. Alexandre Alvaro

Bibliografia

1. Agile practice. 2. User eXperience. 3. User Story. 4. Navigational Distance.  
I. Orientador. II. Universidade Federal de São Carlos. III. Título

---





UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências em Gestão e Tecnologia  
Programa de Pós-Graduação em Ciência da Computação

---

**Folha de Aprovação**

---

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Abner Cleto Filho, realizada em 27/03/2020:

---

Profa. Dra. Luciana Aparecida Martinez Zaina  
UFSCar

---

Profa. Dra. Sabrina dos Santos Marczak  
PUC-RS

---

Prof. Dr. Alexandre Alvaro  
UFSCar

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Luciana Aparecida Martinez Zaina Sabrina dos Santos Marczak, Alexandre Alvaro e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ão) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.

---

Profa. Dra. Luciana Aparecida Martinez Zaina







# Acknowledgements

Special acknowledgment and gratitude expression to,

My parents Abner Cleto and Rosmara Sanches Cleto, who always supported me in everything I decided to do and to always be with me at all times. Thank you for your support, encouragement and affection.

My sister Amanda Cleto, to the support and patience through all my study period.

My advisor Professor Dr. Luciana A. M. Zaina for having welcomed me as a student. Thank you for your confidence, motivation, inspiration, opportunity, and for every thing you taught me through this period as a teacher and as an advisor.

The evaluation board composed by teachers Dr. Alexandre Alvaro and Dr. Sabrina dos Santos Marczak, for having accepted to participate and to contribute to this project.

To all my colleagues of UXLeris Research Group for supporting me with ideas and suggestions.

The company which allowed me to collect and use data for this study, as well as practices and experience I have learned while in there.

I also thank the financial support of the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. I also thank the grant #2017/03397-0 São Paulo Research Foundation (FAPESP).



*I never guess.  
It is a capital mistake to theorize before one has data.  
Insensibly one begins to twist facts to suit theories,  
instead of theories to suit facts.  
(Sherlock Holmes - By. Arthur Conan Doyle)*



# Abstract

Agile practices are approaches widely used in industrial context. User Stories (US) are valuable artifacts to agile teams, being a succinct description of a requirement with its details complemented by other artifacts. User eXperience (UX) is an important cross-cutting quality requirement that has gained spotlight over the past years. Regarding the software development teams structure, the usage of teams that are not co-located is also a practice that has been utilized over the years, gaining more adepts over the years. Remote working has challenges reported in the literature, be it the time lapse among the team members, difficult on communication when compared to a face-to-face approach, and others. The usage of virtual environments and artifact-based communication are encouraged in the literature, given such are common to all team members, whether they are co-located or not. Many approaches on merging agile and UX are presented in the literature, but many issues are still faced in such enterprise. Problems on communication among UX specialists and agile developers, loss of big-picture, information traceability, and others, are recurring in the literature. Studies encourage the usage of artifacts to mediate communication, as well as to hold and share information, but few have analysed how UX information and USs are related in agile virtual environments. This master project presents an investigation on the navigational distances between UX information and USs in virtual environments. The investigation was performed through a case study in a software company and a qualitative analysis on the data collected. The investigation had the goal to understand how agile practitioners related UX information to USs. To do this, we conducted a qualitative analysis in 13 requirement documents of three different industry projects and we also explored the USs derived from such documents. Using the case study outcomes, we compare the issues found in it with the the literature. We identify the ways the artifacts can be related in the virtual environments, describing and providing pros and cons for each connection type. We propose a classification for navigational distances found among UX information and USs. Moreover, we propose a category on the navigational effort required to perform the navigation in virtual environments. The navigation and effort categories proposed extend the work of [Bjarnason et al. \(2016\)](#) on regards to navigational distance, considering it in virtual environments. Our work also contributes to motivate agile teams in rethinking about different forms to organize the UX information in virtual environments. We propose an arrangement template considering virtual environments, where the template displays a way to create USs and link it with UX elements, considering the navigational distance and effort classifications previously mentioned along with the goal to reduce the final navigational distance present among UX information and USs.

**Keywords:** Agile practice, User eXperience, User Story, Virtual Environments, Navigational Distance.



# List of Figures

Figure 1 – Research proposal . . . . .	25
Figure 2 – Fundamentals and Related Work . . . . .	29
Figure 3 – Management tools used across agile vs. waterfall . . . . .	33
Figure 4 – Elements of User Experience . . . . .	36
Figure 5 – Classification of used artifacts . . . . .	41
Figure 6 – Case Study . . . . .	49
Figure 7 – Steps of the analysis . . . . .	51
Figure 8 – Coded text chunk . . . . .	57
Figure 9 – Coding process codes . . . . .	58
Figure 10 – User Story coding sample . . . . .	59
Figure 11 – Task example . . . . .	60
Figure 12 – Document coding mapping . . . . .	61
Figure 13 – User Story coding mapping . . . . .	65
Figure 14 – Venn Diagram - Project 1 . . . . .	66
Figure 15 – Venn Diagram - Project 2 . . . . .	66
Figure 16 – Venn Diagram - Project 3 . . . . .	66
Figure 17 – Coding Grouping - Venn Diagrams . . . . .	67
Figure 18 – Case Study . . . . .	69
Figure 19 – UX Elements dispersion example - project 1 . . . . .	71
Figure 20 – UX Elements dispersion example - project 2 . . . . .	72
Figure 21 – UX Elements dispersion example - project 3 . . . . .	72
Figure 22 – Navigational Distance Analysis . . . . .	75
Figure 23 – UX Elements dispersion sample diagram . . . . .	80
Figure 24 – Navigational Effort . . . . .	86
Figure 25 – Template proposal . . . . .	92
Figure 26 – User Story static template for information organizationn . . . . .	94
Figure 27 – Virtual Environment Step 1 - Epic and Main Repository page creation	96
Figure 28 – Virtual Environment Step 2 - User Story and UX Task creation . . . .	97
Figure 29 – Virtual Environment Step 3 - Specific Repository page creation . . . .	97
Figure 30 – Virtual Environment Step 4 - User Story enrichment and link to UX repository . . . . .	98
Figure 31 – Virtual Environment - User Story link to UX central repository . . . .	99
Figure 32 – Specific Repository - Epic page . . . . .	101
Figure 33 – Specific Repository - UX page . . . . .	101
Figure 34 – UX Central Repository . . . . .	103
Figure 35 – Distance diagram of proposed organization . . . . .	105

Figure 36 – Results propagation . . . . .	115
Figure 37 – Document 1 - User Story 1 coding . . . . .	154
Figure 38 – Document 1 - User Story 2 coding . . . . .	155
Figure 39 – Document 2 - Epic coding . . . . .	158
Figure 40 – Document 2 - User Story 1 coding . . . . .	159
Figure 41 – Document 2 - Confluence 1 coding . . . . .	160
Figure 42 – Document 2 - Confluence 2 coding . . . . .	160
Figure 43 – Document 3 - User Story 1 coding . . . . .	162
Figure 44 – Document 3 - User Story 2 coding . . . . .	163



# List of Tables

Table 1 – Main related works. . . . .	47
Table 2 – Related Work Comparison. . . . .	48
Table 3 – Summarization of Issues found out . . . . .	76
Table 4 – Types of Navigational Distances. . . . .	82
Table 5 – Types of Navigational Effort. . . . .	84
Table 6 – Good practices recommendation. . . . .	87



# List of abbreviations and acronyms

ALM	Application Life cycle Management
ASD	Agile Software Development
AUCD	Agile User-Centered Design
HCI	Human-Computer Interaction
LDUF	Little Design Up-Front
PO	Product Owner
QA	Quality Assurance
RE	Requirement Engineering
SDLC	Software Development Life Cycle
SDUF	Some Design Up-Front
UC	Use Case
UCD	User-Centered Design
UI	User Interface
US	User Story
UX	User eXperience
UXD	User eXperience Design
XP	eXtreme Programming
WIP	Work In Progress



# Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>21</b>
1.1	Main Goal and Objectives	23
1.2	Methodology and Organization	24
1.3	Contributions	27
1.4	Dissertation organization	27
<b>2</b>	<b>FUNDAMENTALS AND RELATED WORK</b>	<b>29</b>
2.1	Agile Practices	29
2.2	User Story	31
2.3	Virtual Environments	32
2.4	User Experience	34
2.5	Elements of User Experience	36
2.6	The theory of distances	38
2.7	Related Work	39
2.7.1	User eXperience and Agile practices	39
2.7.2	Virtual Environments	43
2.7.3	Navigational Distance	45
2.7.4	Conclusions and Related Work Comparison	45
<b>3</b>	<b>CASE STUDY</b>	<b>49</b>
3.1	Context settings	49
3.2	Analysis approach	51
3.2.1	Artifacts Uncovering	54
3.2.2	Documents Coding	54
3.2.3	User Stories Coding	59
3.2.4	Coding Results	61
3.2.5	UX Information Dispersion	68
3.2.5.1	UX Elements Dispersion Analysis	70
3.2.5.2	UX Information Dispersion Findings	72
3.3	Case Study Conclusions	74
<b>4</b>	<b>IMPROVING NAVIGATIONAL DISTANCE</b>	<b>75</b>
4.1	Practice and Theory Comparison	76
4.1.1	Loss of Big Picture	76
4.1.2	UX elements traceability	77
4.1.3	UX elements in agile practices	78

4.1.4	UX elements dispersion . . . . .	78
4.1.5	Navigational distance to UX elements . . . . .	79
<b>4.2</b>	<b>Navigational distance classification . . . . .</b>	<b>80</b>
<b>4.3</b>	<b>Navigational effort classification . . . . .</b>	<b>84</b>
<b>4.4</b>	<b>Reducing the navigational distance and effort . . . . .</b>	<b>87</b>
4.4.1	Good practices recommendation . . . . .	87
4.4.2	Template proposal for information arrangement in virtual environments . . .	92
4.4.2.1	US template . . . . .	93
4.4.2.2	UX Elements arrangement in virtual environment . . . . .	95
4.4.2.3	Repository template . . . . .	99
4.4.3	Template usage in different Tools . . . . .	103
4.4.4	Recommendations considerations . . . . .	104
<b>4.5</b>	<b>Final Considerations . . . . .</b>	<b>106</b>
<b>5</b>	<b>CONCLUSION . . . . .</b>	<b>111</b>
<b>5.1</b>	<b>Contributions . . . . .</b>	<b>113</b>
<b>5.2</b>	<b>Results Propagation . . . . .</b>	<b>115</b>
<b>5.3</b>	<b>Study Limitations . . . . .</b>	<b>116</b>
<b>5.4</b>	<b>Future Work . . . . .</b>	<b>117</b>
	<b>Bibliography . . . . .</b>	<b>119</b>
	<b>APPENDIX A – CODING OF DOCUMENT 1 . . . . .</b>	<b>125</b>
	<b>APPENDIX B – CODING OF DOCUMENT 2 . . . . .</b>	<b>131</b>
	<b>APPENDIX C – CODING OF DOCUMENT 3 . . . . .</b>	<b>141</b>
	<b>APPENDIX D – USER STORY CODING OF DOCUMENT 1 . . . . .</b>	<b>153</b>
	<b>APPENDIX E – USER STORY CODING OF DOCUMENT 2 . . . . .</b>	<b>157</b>
	<b>APPENDIX F – USER STORY CODING OF DOCUMENT 3 . . . . .</b>	<b>161</b>
	<b>APPENDIX G – ARTIFACTS DISPERSION . . . . .</b>	<b>165</b>
<b>G.1</b>	<b>Artifacts Dispersion - Document 1 . . . . .</b>	<b>165</b>
<b>G.2</b>	<b>Artifacts Dispersion - Document 2 . . . . .</b>	<b>168</b>
<b>G.3</b>	<b>Artifacts Dispersion - Document 3 . . . . .</b>	<b>172</b>

# 1 Introduction

Agile software practices have brought several advantages to software development such as having more precise requirements due to a reduced scope, better work effort estimation, early feedback due to frequent deliveries, and others (PETERSEN; WOHLIN, 2009). However, only agile practices do not cover all the needs encountered during the project development cycle, such as the usability of the software product (JURCA; HELLMANN; MAURER, 2014).

During the software development life cycle there is a need to convey to developers the functional aspects (processes, business rules, user interactions, and others) that constitute the development process. Surveys carried out through research and reviews of the literature reveal that communication and requirement documentation in agile projects still present challenges (HESS; DIEBOLD; SEYFF, 2017). Such challenges can be a reflection of the agile practice's structure, which, according to the agile manifesto, does not have a great focus on the documentation itself (BECK et al., 2001). Agile practices also may have communication issues between the stakeholders involved in software development, such as clients, users, analysts, developers, and others. Methods and techniques for improving communication on software development are scarce, and it may be caused due to divergences from the point of view, knowledge, experience, needs, vocabulary or the time available to each person involved (BJARNASON; SHARP; REGNELL, 2019; JURCA; HELLMANN; MAURER, 2014; TAIBI et al., 2017).

According to Hassenzahl and Tractinsky (2006), in the technology ever-changing environment, interactive products became not just only useful and usable, but also fascinating and fashionable desired things. In such environment, User eXperience (UX) has gained the spotlight, especially in the Human-Computer Interaction (HCI) field (HASSENZAHL; TRACTINSKY, 2006). Don Norman and Jakob Nielsen (2013) defines UX as including all aspects of end-user interaction with the company, its services, and its products.

Brhel et al. (2015) already presented the importance of Agile Software Development (ASD) and UX, concluding that both have become main features in their respective fields. Moreover, given the need to deliver business value to the customer in a fast-changing environment, taking into account the needs of end-users, the integration of ASD and UX seemed to be a promising enterprise. Attempts on integrating agile and UX practices have been done, but problems on such process are still to be solved (BRHEL et al., 2015).

Several strategies are proposed and employed by different development teams to integrate agile practices and UX principles. Applying design upfront techniques, in which

UX practitioners work ahead of the actual software development have been proposed (SILVA et al., 2011; GARCIA; SILVA; SILVEIRA, 2017) but already discarded, given the merging of UX and agile seems to be a more realistic scenario (GARCIA; SILVA; SILVEIRA, 2019). This indicates that both UX and agile areas should work as close as possible, instead of separated (GARCIA; SILVA; SILVEIRA, 2019). Other works present scenarios in which developers and UX practitioners work together in initial stages of a project but later they split. In such scenario, UX practitioners work closer to the users, getting design approvals from them but without developer's input (PLONKA et al., 2014).

Other works also present solutions for integrating UX and agile, but issues are reported in several studies. Among such issues, it is possible to highlight the communication issue on the ones involved in the project, as well as information loss during the project life cycle (BRHEL et al., 2015; BUDWIG; JEONG; KELKAR, 2009; SILVA et al., 2011). Studies also highlight that some artifacts used during the development life cycle have as main purpose to serve as communication and as a source of information for those involved, facilitating the search and sharing of information (GARCIA; SILVA; SILVEIRA, 2017; HESS; DIEBOLD; SEYFF, 2017; BRHEL et al., 2015).

Among the various artifacts adopted in agile practices, the User Story (US), is widely used (BIK; LUCASSEN; BRINKKEMPER, 2017; BRHEL et al., 2015; SOARES et al., 2015; LUCASSEN et al., 2015). According to Choma, Zaina and Beraldo (2016), a US describes a functionality that aims to deliver something of value to the user or to the customer, helping the development team make decisions based on the information it has at its disposal, over the duration of the project.

However, the process of creating USs is not necessarily straightforward and can be coordinated according to the situations and the teams they are used. Besides, only USs may not contain all the information necessary for the complete development of a given product (HESS; DIEBOLD; SEYFF, 2017).

Difficulties have also been reported regarding the communication of non-functional aspects in agile practices. This can be related to the writing of US and the aspects of UX present in it (LUCASSEN et al., 2015). Lopes et al. (2017) investigates how software developers use HCI techniques and methods in US writing, concluding through a qualitative analysis of writing, that there was no further detailing of the UX aspects present in the US.

In addition, given the difference of artifacts used in agile practices and by UX specialists (GARCIA; SILVA; SILVEIRA, 2017), we can have an indicator that information not present within USs are placed in other artifacts. Such artifacts may be spread across different platforms used by the teams involved in a project. Such scenario could lead to the increase of the navigational distance when UX and agile practices are applied together.



Project teams composed of remote members are becoming a common practice, although the co-located teams configuration is still used. Virtual project teams have been defined as groups of people who are not co-located, using virtual environments, such as Trello, Confluence and Jira (later described in section 2.3), to work together to accomplish a goal (REED; KNIGHT, 2010). The information for the remote worker is limited to the virtual environment (DESHPANDE et al., 2016), reinforcing the importance of virtual environments, that may also be used by co-located teams, given that they facilitate information sharing, tracking and management, when compared to the physical space, which can still be used along with the virtual environment. With virtual environments usage, the presence of virtual artifacts is also made. Placing artifacts, which are information holders (DESHPANDE et al., 2016), into the virtual environment relates to the work of Bjarnason et al. (2016), which explores the distances present among tasks that constitute the development of the system.

Among the distances presented by Bjarnason et al. (2016), the navigational distance can be understood, in a virtual environment, as the number of clicks required to go from a piece of information to another (BJARNASON; SHARP, 2017). The work of Bjarnason and Sharp (2017) removed the navigational distance from a further study on the other distances (BJARNASON et al., 2016). Therefore, the navigational distance is a topic still to be further analyzed in the literature, especially if considering virtual environments. Another topic to be further investigated is the distances between UX information and agile artifacts, in particular, the US.

## 1.1 Main Goal and Objectives

This master project aimed to verify which UX elements are used along with agile artifacts, focusing in the US. Such goal was achieved having the sub-topic of virtual environments. This was due to all artifacts studied, as well as its relations, were stored in virtual tools.

Once understanding which UX elements used with USs in virtual environments, another goal of this project was to understand how such UX elements relate, in a virtual environment, to USs. After analyzing how such relation is made, the analysis of navigational distance was performed. The project also had the goal to propose templates to relate UX information with USs in virtual environments, having the navigational distance analysis as the basis for such templates design. The templates aim to decrease, or limit, the maximum navigational distance required to find UX information, having the US as the starting point of the navigation.

These goals aim to bring insights on how to better arrange and relate the UX elements with the USs in the virtual environments. From these goals, the following Research

Questions (RQ) have been outlined for this project:

**RQ1: How are UX information and USs connected into software virtual environments?**

**RQ2: What are the navigational distance found to access UX information from USs into software virtual environments?**

Considering the RQs, as well as the main goals, the following specific objectives were also proposed:

- To create a classification system to the types of relationship the artifacts can have in a virtual environment. Such classification aimed to facilitate the understanding of the artifacts relation as well as to provide insights on how to better arrange them.
- To create a classification for the navigational distances, as well as for the effort required to perform it into virtual environments. The classification had the goal to bring understanding on how the artifacts arrangement into virtual environments can impact the users. Such classifications are done considering a developer's point of view. They also expand the work on distances of [Bjarnason et al. \(2016\)](#), [Bjarnason and Sharp \(2017\)](#), giving new insights to the navigational distances.

## 1.2 Methodology and Organization

To achieve the outlined objectives, this master project was organized in order to raise issues on Agile-UX integration currently reported in the literature. Next, an industry case study was executed to understand how UX information could be found in agile projects. The dispersion of UX information in virtual environments was also a topic covered in the case study. In order to contribute on decreasing the navigational distances, templates for US creation, its relation with UX information and its arrangement into virtual environments were proposed. Figure 1 represents the 6 main steps to conduct this study.

The case study started with the analysis of requirements documents belonging to three different real projects. These projects belong to a company that develops software for the financial market. These documents are used as input for writing USs. For the achievement of this proposal, the principles of agile software development practices as well as the main aspects of UX were taken into account, having as reference the UX elements framework proposed by [Garrett \(2010\)](#).

**Step A - State of the Art Investigation:** The bibliographic research involved the search for the state of the art in the areas of software development, with a focus on agile practices, and on UX practices, addressing the artifacts used in each of the areas.

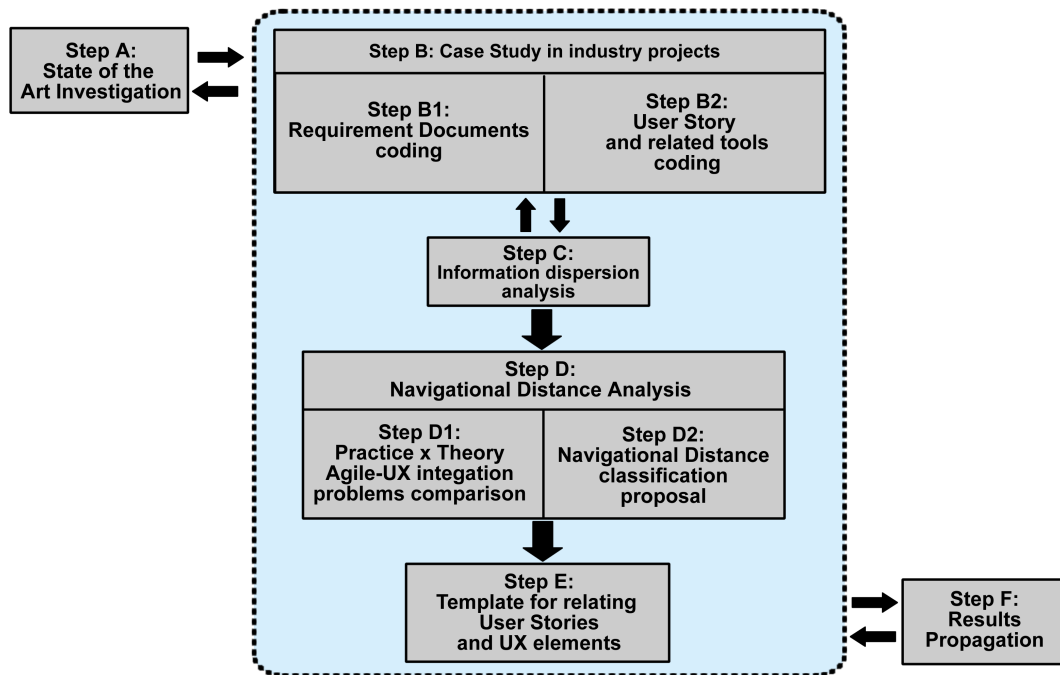


Figure 1: Research proposal

Source: Author

The focus was given to search for the current UX-Agile integration techniques, aiming to find the main problems as well as the available solutions. This step took place throughout the entire project and resulted in a state of the art bibliographic review document where: (i) there is the use of UX practices with agile practices in the same project; (ii) the main problems; and (iii) the main artifacts in each area.

**Step B - Case Study in industry projects:** A qualitative analysis was carried out in the requirements documentation used by three real projects that use agile development practices. As company data is confidential, the data presented in this project has been transformed without losing the essence of the information.

Step B1 - Requirement Documents coding: Given the three projects' requirements documents and based on the framework proposed by [Garrett \(2010\)](#), as well as the UX elements previously found in Step A, the requirements documents were mapped to find which elements and artifacts of UX were present, explicitly or implicitly, in the documents. Such mapping was carried out using coding techniques ([STRAUSS; CORBIN, 1990](#)), which assign labels to parts of a text aiming to classify them to provide future analysis. The result of this step resulted in a table with the mapping of all UX elements and artifacts used in the requirement documents.

Step B2 - User Story coding: Considering the mapping of the artifacts (Step B1), the same mapping was performed in USs, which were written based in the requirement documents. This step checked which UX elements were used in the requirement documents

and also in the USs. This step evaluated if the USs contained the aspects of UX previously found in Step B1. Given that USs could contain internal and external relations (i.e. US be linked to another US or Task within the same tool or point to another tool), the coding on related places linked to the US was also performed. The same coding principles used in step B1 were used in this step.

**Step C - Information dispersion analysis:** With the information coded from Step B, an analysis of its dispersion in the virtual environments used in the projects was done. The dispersion was based on how distant the information was from one another, being USs the initial point for such dispersion verification. As mentioned in step B2, some UX elements could be placed outside a given US, but still be linked to it. Therefore this step aimed to analyze how the UX information previously coded could be found in the virtual environments, having the US as the initial point for such arrangement analysis.

**Step D - Navigational Distance analysis:** Given the information dispersion analysis done on Step C, we performed an analysis considering the navigational distances ([BJARNASON et al., 2016](#); [BJARNASON](#); [SHARP, 2017](#)) to understand how the distances found could then be classified.

Step D1 - Practice x Theory Agile-UX integration comparison: With the issues reported in the literature (step A) on integrating Agile and UX, a comparison with the problem and the scenarios studied in this master project was done, showing the main problems that could be verified in the projects studied.

Step D2 - Navigational Distance classification proposal: Once the navigational distances were analyzed, a classification to them was proposed, based on the type of action and the effort required to perform such navigation into virtual environments. Such classification is a contribution to the description of the navigational distance exposed by [Bjarnason et al. \(2016\)](#) and [Bjarnason and Sharp \(2017\)](#).

**Step E - Elaboration of recommendations and templates for User Story and UX elements relation:** Elaboration of recommendations and templates for the US writing. Analysis of the practices and patterns found in the projects and proposal for using both software development artifacts with UX information. The proposal will aim at reducing the navigational distance ([BJARNASON et al., 2016](#)) between US and UX elements in virtual environments, without the loss of information contained in them.

**Step F - Results propagation:** Consists of the compilation and synthesis of all the material studied and analyzed in this project for the dissemination of appropriate knowledge, consisting of the writing of the dissertation to be defended and the production of articles to be published.

## 1.3 Contributions

From the execution of all the steps previously proposed and aiming to meet the study goals, the contributions of this research embrace:

- An industry case study on three projects analysing how UX information relates to USs in virtual environments;
- Navigational distance mapping between UX information and USs;
- Description on the types of relationships UX elements can have with USs considering virtual environments;
- Categorization of navigational distances, expanding the work of [Bjarnason et al. \(2016\)](#), [Bjarnason and Sharp \(2017\)](#);
- Categorization of effort to perform navigation in virtual environments, expanding the work of [Bjarnason et al. \(2016\)](#), [Bjarnason and Sharp \(2017\)](#);
- Proposal of recommendations to better relate UX information and User Stories into virtual environments, aiming to reduce the navigational distance between them;
- Templates for User Story and enrichment process through the creation of specific UX tasks under the User Story;
- Templates of Specific and Central Repositories to hold UX information, which will be used as a central location to navigate from US in order to retrieve UX information.

Also, during the development of this project, an article entitled "Navigational distances between UX information and User Stories in agile virtual environments" was accepted on ICEIS 2020. This article does not contain all conclusions presented in this final text, therefore new publications may be done considering the latest outcomes of this project.

## 1.4 Dissertation organization

This dissertation is organized into five chapters containing: introduction; fundamentals and related work; case study and its methodology; recommendations to reduce navigational distance containing navigational distances classification, navigational effort classification, US templates and UX Repositories templates; and conclusions. The current chapter covered the introduction, dissertation goals, contributions and methodology.

Chapter 2 will present some fundamentals used throughout the dissertation, such as agile methodology, user experience and the theory of distances. The chapter will also

present a summary of related works found in the literature that relates to this dissertation, showing a discussion of such works to the current dissertation main subject.

Chapter 3 covers the case study performed, explaining the steps performed in it as well as the results observed from it. Chapter 4 presents a set of issues found in both literature and the case study from Chapter 3. This chapter also proposes recommendations that aim to avoid or decrease the occurrence or severity of some of those issues. The classification of navigational distances, as well as for its effort into virtual environments is also presented in Chapter 3. We also present the templates of US creation and enrichment process with UX information, along with the proposal of UX repositories to be used in virtual environments to centralize UX information.

Chapter 5 concludes the dissertation highlighting the main discoveries of this dissertation and possible future work to be done, considering the contributions of this work and the validations that need to be done in order to give it reliability.

## 2 Fundamentals and Related Work

This chapter will summarize the main concepts of agile practices, UX, navigational distance, virtual environments, covering virtual management tools mainly used in agile projects. This chapter also presents a summary of the main related work found during project development.

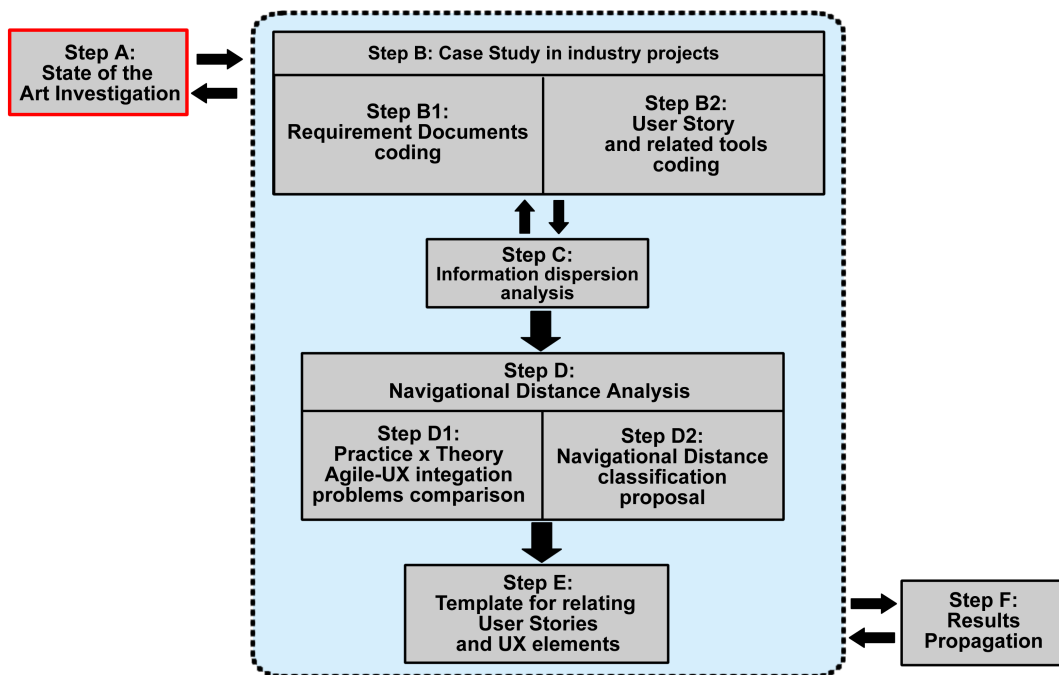


Figure 2: Fundamentals and Related Work

Source: Author

### 2.1 Agile Practices

Agile practices are already the most common software development approaches in the industry. In Agile practices, the focus is on lightweight working practices, constant deliveries, and customer collaboration over long planning periods, heavy documentation, and in flexible development phases (KUPIAINEN; MÄNTYLÄ; ITKONEN, 2015).

The agile requirement specification is not centralized in one phase before development; instead, this activity is evenly spread throughout development (CHOMA; ZAINA; BERALDO, 2016).

The advantage of such a process is that the system is built incrementally, with high-quality adaptive software, being developed by small teams using the principles of

continuous improvement and testing based on fast feedback (DYBÅ; DINGSØYR, 2008). Such process aims to reduce errors that can be easily corrected.

Moreover, lessons learned during development can be applied in the course of the project, as stated in the Agile Manifesto, which encourages fast responding to change over following a plan (BECK et al., 2001). Another principle of the agile practices is that the most efficient and effective method of conveying information to and within a development team is face-to-face conversation (BECK et al., 2001).

Given such principles, it can be assumed that in agile practices, documentation is not the main focus, since the contact with the user tends to be close and constant (BECK et al., 2001). This opens the possibility of software changes being suggested, requiring the development team to be more flexible and adaptive.

Agile practices are also known for their cross-functional teams, which include members from different functional groups who have similar goals. As stated by Bjarnason, Wnuk and Regnell (2011), in agile practices, developers, testers, designers, and others, work as a single team. Such concept helps reducing challenges such as over scoping requirements and communication gaps.

Different ways of implementing the agile practices have been tried; among them, the most popular can be taken as Scrum, eXtreme Programming (XP) and Kanban (MATHARU et al., 2015), which are briefly described below. It is important to notice that for the current study, no specific agile practice will be taken into account, given that the agile practices are not being the focus of this study.

The main characteristic of Kanban is that it provides means to visualize and limit the work in progress during the software development process. It emphasizes on scheduling the work to facilitate product delivery. Kanban Board allows easy visualization of current work, productivity maximization, continuous delivery, waste minimization and limit of work in progress (WIP) as its main characteristics (MATHARU et al., 2015).

The Scrum methodology has the cycles or iterations, called sprints, as its main characteristic. A product deliverable may be release at the end of each cycle, providing continuous delivery in time boxes. Each sprint contains all phases of the software development, e.g. design, development, testing, user acceptance, etc. (MATHARU et al., 2015).

The XP methodology, is based upon twelve principles: Small releases; Planning game; Refactoring; Testing; Pair Programming; Sustainable pace; Team code ownership; Coding standard; Simple design; Metaphor; Continuous Integration and On-site customer. The XP methodology requires a great effort of the whole team, being each team member encompassed of a broad range of skills (COHN, 2004). Extreme Programming is also characterized by its intense levels of interaction with customers during the software development process (MATHARU et al., 2015).



Despite all advantages of agile practices, works such as [Garrett \(2010\)](#) and [Inayat et al. \(2015\)](#) point that agile practices still struggle on working with requirements that focus on system quality, including its internal quality, i.e. maintainability, testability and external quality usability. Such requirements are known as non-functional requirements (NFRs). Neglecting such NFRs is considered a major challenge for agile practices and can be the reason for massive lapse and rework ([INAYAT et al., 2015](#)).

## 2.2 User Story

Among several artifacts available in software development process, User Stories (US) are the most popular requirements notation in agile projects ([BIK; LUCASSEN; BRINKKEMPER, 2017](#)).

[Cohn \(2004\)](#) describes US as a functionality that will be valuable to either a user or purchaser of a system or software. User stories can be summarized by its three main aspects:

- a written description of the story used for planning and as a reminder;
- conversations about the story that serve to flesh out the details of the story;
- tests that convey and document details and that can be used to determine when a story is complete.

The process of writing US does not need to follow an strict rule, but [Cohn \(2004\)](#) proposes the following template: *As a <role>, I want <function> so that <business value>*. Such template covers three elements that can be labeled as the role, the goal, and the reason. The three elements of the standard USs template address: **Who** wants the functionality, **What** it is they want and **Why** they want it ([COHN, 2004](#)).

Extensions for such template are reported in the literature. As an example, [Choma, Zaina and Beraldo \(2016\)](#), aiming to incorporate UX aspects into the USs elaboration, proposes the usage of the following template: *As a <Persona>, I want/need <goal>, for this <interaction>, through/when <task/context>*. The template also covers acceptance criteria an US evaluation, by adding in its writing the following: *I evaluate that my goal was achieved when <feedback>*. ([CHOMA; ZAINA; BERALDO, 2016](#))

Along with USs, it is also important to describe the Use Case (UC), as this will also be mentioned in this master project. According to [Cohn \(2004\)](#), UC is a generalized description of a set of interactions between the system and one or more actors, in which an actor is a user or another system. Use cases can be written in unstructured text or to fit a structured template. The main difference between USs and Use Cases can be considered as their scope. The USs are kept small due to their use in the Agile process, which restrict

development time to shorter periods given constant deliveries. The UC covers a larger scope compared to US, which leads to the tendency to consider a US as one of several scenarios that constitute an UC (COHN, 2004).

Given the US characteristics, it can be written to cover several functionalities at once, such scenario leads to the need to decompose the US. This is done given a large task might not be delivered within the short period of time available in an agile project. One of the agile characteristics is its constant delivery, which occurs in small time frames. Such characteristic enable only the development of small items to be done or preferred. The act of breaking down a US into smaller items is called decomposition (TAIBI et al., 2017).

Breaking down USs is usually done in a way that its parts have a scope large enough to provide customer value, but small enough that the effort to implement such a story can be estimated with a low risk. Another relevant factor is that a smaller scope is likely to make the US less complex compared to one with a larger scope, and the fact that possible problems, such as the emergence of unknown details, inadequate minimized (TAIBI et al., 2017).

## 2.3 Virtual Environments

In this master project, virtual environments, is a term which describes the virtual platforms, tools, workspaces, etc., used by the ones involved in a project. Different from physical locations, virtual environments provide ways for users to store, relate, find and communicate information. The usage of virtual environments is especially present in remote teams, which is a common practice in nowadays fast-changing project development scenario (DESHPANDE et al., 2016), but its usage is also done in co-located teams, one that such platforms aim to facilitate the information sharing and management for the team members.

Among different artifacts, practices and methodologies available in the software development lifecycle (SDLC), the concept of Application Lifecycle Management (ALM) is an important part of the processes. According to Kassab (2014), tools are able to manage ALM activities from initial project concept, through requirements analysis, development, and testing. Deshpande et al. (2016) explores the importance virtual environments, specially for non co-located team members, concluding that such environment are the key place for information sharing in distributed teams, which is a practice that is becoming common over the year (REED; KNIGHT, 2010; DESHPANDE et al., 2016).

To provide support of both co-located and remote teams, several tools that help in the SDLC and ALM management are available. The study done by Kassab (2014) compares the usage of ALM tools in agile and waterfall methodologies. Figure 3 shows the difference of management tools used in Agile and Waterfall practices. We can see that the

usage of JIRA<sup>1</sup> is presented as most common among Agile practices.

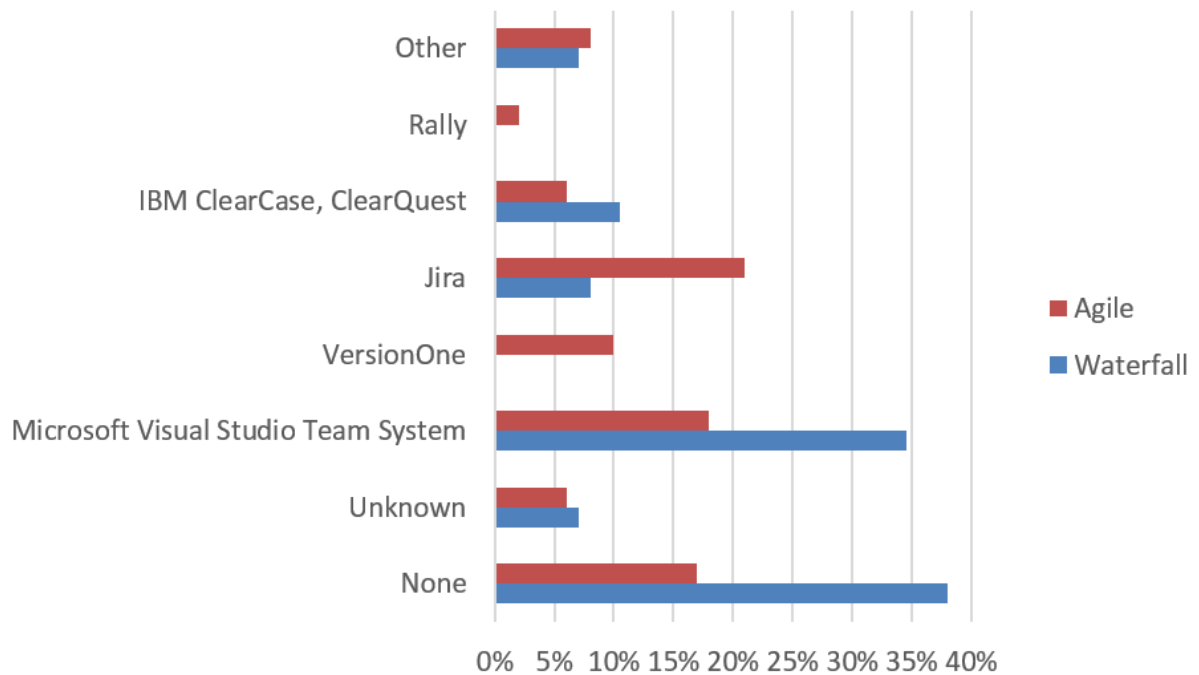


Figure 3: Management tools used across agile vs. waterfall

Source: Reproduced from [Kassab \(2014\)](#)

Jira can be described as a project tracking system ([DESHPANDE et al., 2016](#)) that allows project stakeholders to create cards with task or activities descriptions, being such cards called tickets. In the study of remote working done by [Deshpande et al. \(2016\)](#), it was concluded that a key project-related artifact used for tracking purposes was the ticket. This supports the study of [Bik, Lucassen and Brinkkemper \(2017\)](#), reinforcing the importance of USs.

Another application similar to Jira is Trello<sup>2</sup>, which allows the creation of boards, lists and cards (may it be structured like an US or not), in order to organize and prioritize the project tasks.

Both applications are similar in its core, with few differences, which are reported in the work of [STOPA and RACHID \(2019\)](#) as: Trello offers task lists that are simple, it is a task and project management tool that is easy to use and fits into small/medium size companies, or big companies that don't demand many reports. Jira offers a wide management and traceability tool for small to big projects, enabling the users the generate elaborated reports if necessary.

Confluence<sup>3</sup> is a tool that allows virtual pages to be created into a structured and

<sup>1</sup> JIRA: <https://www.atlassian.com/software/jira>

<sup>2</sup> Trello: <https://trello.com/>

<sup>3</sup> Confluence: [www.atlassian.com/software/confluence](http://www.atlassian.com/software/confluence)

hierarchical environment. It is distributed by Atlassian<sup>4</sup>, which is also the owner of Jira. The pages create in Confluence tend to create a wiki-based collaboration management system.

Similar to Confluence, MediaWiki<sup>5</sup> is a free, open-source wiki software that allows the creation and editing of dynamically generated web pages (BERMAN; BARNETT; MOONEY, 2012).

SharePoint<sup>6</sup> is a web application, which works as a hub to information centralization (i.e. a repository) through collaboration, web publishing, and file sharing (BERMAN; BARNETT; MOONEY, 2012).

Many other applications are available that provide similar functionalities to the ones here mentioned. This master project though, was focused on Jira, Confluence and SharePoint, which are the applications used in the projects of this projects case study. Although differences exist among applications and its concurrent, the conclusions presented in this master project didn't consider any specific characteristic of any tool. We also believe that the conclusions later exposed in this master project can be applied in any tool, although some changes need to be done to better fit into the particularities of each tool.

## 2.4 User Experience

According to Law et al. (2009), UX results from the wide range of potential benefits that users can obtain from a product; and can be classified as something new, that must be an integral part of the HCI domain, supported by the User Centered Design (UCD) practices (LAW et al., 2009).

According to ISO/IEC 9241-210, UX can be understood as the *"perceptions and responses of the person (user) resulting from the use and/or anticipated use of a product, system or service"* (ERGONOMICS... , 2010). Don Norman and Jakob Nielsen (2013) defines UX as *"encompassing all aspects of the end-user interaction with the company, its services, and its products"*.

According to Garrett (2010), UX is the experience created from a products usage. When a product is being generated, one has to pay close attention to what it does; however, the user's experience is often the neglected side of the equation of how something works, which can be the difference between a product of success and failure.

User-Centered Design (UCD) is described by Vredenburg et al. (2002) as a multidisciplinary design approach, which is based on actively involving users for a clear understanding of user and task requirements, and the iteration of design and its evaluation.

<sup>4</sup> Atlassian: <https://www.atlassian.com/>

<sup>5</sup> MediaWiki: <https://www.mediawiki.org/wiki/MediaWiki>

<sup>6</sup> SharePoint: <https://products.office.com/sharepoint>

UCD can also be considered a key point to a product usefulness and usability. According to [Garrett \(2010\)](#), UCD ensures that the usability aspects, found and met in the development process, do not happen by accident. This would be given that, thinking about the user experience, dividing it into elements and looking at them from various perspectives, it is possible to ensure that all ramifications of decisions are made intentionally.

Many techniques, according to [Garcia, Silva and Silveira \(2017\)](#) and ([GARCIA; SILVA; SILVEIRA, 2019](#)), can be applied to develop a better experience to users, such as Personas, Sketches, Scenarios, and Wireframes, Prototype, Mockup, and others ([ROGERS; SHARP; PREECE, 2019](#)). When integrating UCD and agile practices, [Schön, Thomaschewski and Escalona \(2017\)](#) states that some UX artifacts can be used for communication, elaboration, validation, and documentation of requirements in agile environments, reinforcing the importance of such artifacts. Also, in Garrett's framework (Figure 4), some of such UX artifacts are mentioned as a possibility of integrating such in the project development life-cycle ([GARRETT, 2010](#)).

Many techniques are used in UX, below we present some of the most used ones ([GARCIA; SILVA; SILVEIRA, 2019](#)).

**Prototyping:** An application's interface (also know as User Interface - UI) can be difficult to interact with, causing performance issues. Such problem can be the result of a poorly designed UI. One of the practices that aims to make such scenario less present is the usage of prototyping during the early stages of a software development. According to [Wilson and Rosenberg \(1988\)](#), usability issues are difficult to repair, especially those detected in project final stages or after the system has already been put into operation. A prototype, according to [Rogers, Sharp and Preece \(2019\)](#), can be anything from a paper-based storyboard through to a functional software, which aims to allow stakeholders to interact with an envisioned product. Prototypes are a useful when discussing ideas with stakeholders, helping in the communication that, sometimes, may be difficult given the different points of view or knowledge are different among development team members and final users; also being an effective way for designers to explore design ideas, given that proposed changes can be easily applied into such prototypes ([ROGERS; SHARP; PREECE, 2019](#)).

**Scenario:** The definition of Scenarios can be stated as informal stories about user tasks and activities. Scenarios can be used to model existing work situations, but they are commonly used to express proposed, or imagined, situations aiming to help in conceptual design ([ROGERS; SHARP; PREECE, 2019](#)). [Wilson and Rosenberg \(1988\)](#) makes a relationship among Scenarios and UC, stating that a (user) scenario is a sequence of actions, representing a combination of UC that are done in order to achieve an specific goal. Such scenarios can be represented in a narrative form or even into a flow diagram. Scenarios can be used as scripts for user evaluation of prototypes, as the basis of storyboard

creation (ROGERS; SHARP; PREECE, 2019).

**Persona:** Persona is a rich descriptions of typical system users, representing the different end-user profiles. A persona does not describe a real person, but are realistic enough to create a sense of reality in the same. A persona usually represents a number of real users who have been involved in data gathering. Among its components, a persona will include the goals, description of the user's skills, attitudes, tasks, and environment in which the persona will be in (ROGERS; SHARP; PREECE, 2019).

## 2.5 Elements of User Experience

In the book *Elements of user experience, the: user-centered design for the web and beyond*, Garrett (2010) describes the elements of UX proposing a process of software creation from the UX perspective. This process aims to ensure that no aspect of the user's experience with the product happens without it being conscious or with explicit intent (GARRETT, 2010). The process is carried out by five elements, each one being a plane in the process framework (Figure 4). The five elements (or planes) are: Surface, Skeleton, Structure, Scope and Strategy. The following is a brief description of each of the planes elaborated by Garrett (2010).

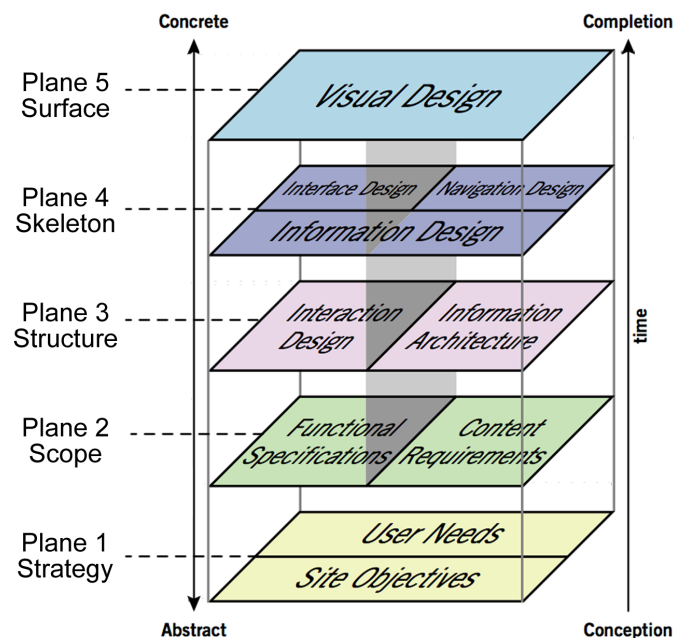


Figure 4: Elements of User Experience

Source: Adapted from Garret (2000)

**Strategy plane:** In this plane, it is discussed what the application to be developed will or will not cover, either from the point of view of the customer or the user. For example,

from the point of view of the company (client), the product (system), should sell products, and make new clients to become recurring clients, displaying products according to their interest. From the user's point of view, the action to be performed in the system is the purchase of the product. The strategy plane also covers other aspects, such as advertising. This being the initial plane, it is assumed that the main aspects, functionalities and objectives of the development are traced. Also the bases of the application are defined, such as logo, colors and patterns. The objectives, in this plane, should be broad, but still containing meaning, which can be described as big-picture. There is still the definition of target audience, a task that can be performed using methods such as Card Sorting and Persona (GARRETT, 2010).

**Scope plane:** In this plane, the scope itself is defined, covering the system requirements, processes and expected results. The mapping of the framework, connecting and documenting individual requirements, aiming to create the skeleton of the final product is done in the scope plane. In the Scope plane the documentation is written and the prioritization of the requirements is performed, factors that makes the importance of the user be included in the processes is more evident. For this plane, one of the artifacts to be employed would be the Scenario.

**Structure plane:** The definition of the application structure can be understood by its flow, which dictates how each part of the software will be distributed and accessible. This definition is also responsible for making the connection between the features of the application, including what information the users will have access to and how it will be accessed. Hierarchical structures such as horizontal, vertical or organic, alluding to how the features will be arranged in the system, are also part of this plane. In this plane are also focus of analysis issues such as application consistency, employee vocabulary, field nomenclature and menus, among others.

**Skeleton plane:** Once defined the positions of features in the design plane, which can be seen as the pages or screens of the application, the skeleton plane is responsible for dealing with design issues at a more closed level, caring about issues such as the positioning of the elements in a given screen, aiming to perfect such positioning to guarantee maximum effectiveness and efficiency, with a view to making the interface familiar to users. In this plane are also elements such as metaphors (use of icons), navigation and availability of information, such as whether it will be visualized in a textual form or in a graphic. An artifact to be used in this step would be the wireframe, which performs, at a low level, a mapping of the elements present in a given screen and how they are arranged in it.

**Surface plane:** For the surface plane, all the aspects related to the user's feelings and experience when using the application, dealing with design issues at the highest level, considering all the visible components, their layout, aspect, actions, typography, palette of colors, among others. This plane is the most important given that it covers all the others,

representing the final result being presented to the user and, therefore, what will cause it the greatest impact. In this plane having the user close during the development is very important, once that, in this plane, the vision of what will be the final identity of the product being developed is more evident. Although it can be realized in previous planes, it is in this plane that the use of prototypes becomes more important.

The division made by [Garrett \(2010\)](#) is interesting because, from the bottom up perspective, there is no (initial) concern with aspects related to how the application will look at the end (UI aspects, for example), and each plane has a well- defined and isolated meaning, although, as state previously, each plane is connected to the others.

In addition to the planes described here, [Garrett \(2010\)](#) also divides each plane into two parts. One part is responsible for the final product functionalities, the other is responsible for the information. For the functionality, one has the concern with the processes (actions) that users perform, while for the information, aspects such as the availability and visibility of the information are treated. With this, it is possible to divide each of the planes into two distinct but interconnected groups.

## 2.6 The theory of distances

The theory of distances in software engineering, addressed by [Bjarnason et al. \(2016\)](#), categorizes the difference of position or level between the stakeholders, artifacts, or activities. Such position differences, or distances, are related to the effort required to accomplish the tasks that constitute the development of the system. The distances are classified into eight sub-categories:

- Geographic - physical distance of the position between stakeholders;
- Organizational - hierarchy of stakeholders;
- Psychological - attributed to personality or opinion factors of those involved;
- Cognitive - involves aspects of knowledge or level of competence;
- Adherence - analyzes the difference between the specifications in the documents and the final product;
- Semantics - compares the level of similarity between two related artifacts;
- Temporal - difference of time in which two activities are performed;
- Navigational - difference in position between artifacts, for example, the length of the path to navigate between a requirement and the test cases that verify.



For this study, the navigational distance will be used, given that in both UX and agile practices, different artifacts are used (LISKIN, 2015). Among several issues faced when trying to work with both areas in one single project, the communication and loss of big-picture can be highlighted (SILVA et al., 2011; JURCA; HELLMANN; MAURER, 2014; SCHÖN et al., 2017; HESS; DIEBOLD; SEYFF, 2017; GARCIA; SILVA; SILVEIRA, 2017). Also, managing many different artifacts is also an issue already reported in the literature (LISKIN, 2015). But the usage of artifact for mediating communication is also a practice encouraged (BRHEL et al., 2015; GARCIA; SILVA; SILVEIRA, 2019).

According to Bjarnason et al. (2016), coordination and communication within software development, are affected by distances. Centralizing, or linking, the different artifacts used in a project, could be a way to decrease one of the distances categorize by Bjarnason et al. (2016), the navigational distance, that is caused by the difference in position between artifacts. In the case study conducted by Bjarnason and Sharp (2017) though, the navigational distance was excluded from the analysis. This opens a gap, that this master project aims to cover.

## 2.7 Related Work

The current master project is based on four main subjects: Agile practices, User eXperience, Virtual Environments and Navigational Distance. The bibliographic analysis was done using an ad hoc methodology, using the aforementioned subjects as main keys for the search, which was conducted especially over conferences and journals. The search was done following a snowballing approach, in which the bibliography of each work was also analyzed (WOHLIN, 2014). The analysis of each work was done considering three main steps: i) The work abstract was read, checking whether the same contained any of the main subjects related to this master project; ii) If passing the first step, the text would have its introduction and conclusion read, to filter the ones that do not relate to this master project; iii) The text that passes the second reading process, would have its full content read and, if applicable, be part of the reference work of this master project. If a text passed the mentioned step three, its related work would be searched and pass through the same process described.

Below we present a summary of the main related work of each subject aforementioned. Later, we give a brief discussion of all the related works, also presenting a comparison between them with this master project.

### 2.7.1 User eXperience and Agile practices

The Case Study conducted by Budwig, Jeong and Kelkar (2009), addresses ASD and UX, observing problems of integration between UX teams and ASD teams. Among

such issues, the frequency of changes in design that were not communicated clearly caused confusion among the team. Also requirements that were not well documented, led to confusion about UX deliverable. In the same study, it is also concluded that having UX work being done at the same time (or *sprint*) as the development team, may increase such problems. Therefore the study proposes that they should work in different *sprints*, i.e. the UX work should be done up-front of the actual development.

Garrett (2010), presented a framework called "*The Elements of User Experience*" (see Section 2.5), which relates UX information into five main elements. Each element was then proposed to be used in a software development life-cycle phase. Through such framework, the integration of UX aspects into the project development could be seen as split throughout the project life-cycle.

In a Systematic Literature Review, Silva et al. (2011) presents results on UCD and agile practices. In this review, it is pointed out that in 15 of the studies analyzed, there was evidence that the UX teams should work ahead (in *sprints*) when compared to the development team. Such discovery is also supported on the work of Budwig, Jeong and Kelkar (2009) and the Literature review of Brhel et al. (2015). Silva et al. (2011) also highlighted the loss of big-picture, and teams working in different time-frames, being this called LDUF (Little Design Up-Front) or SDUF (Some Design Up-Front), in which the UX team works ahead of the development team.

In a more recent work though, Silva et al. (2018) states that, the separation between the UX and Agile teams seem to be decreasing, therefore the proposal of the teams working in different time periods is to be revisited. In Silva et al. (2011) work, it is highlighted the US usage, as well as other artifacts, such as Personas and Scenarios, is high among agile and UX practitioners. In this study, the big picture issue is also covered, being this one of the recurring problems mentioned in the literature, were the usage of multiple artifacts may be one of the causes of the Big Picture view loss, once that the information could be spread among the artifacts.

The review of Silva et al. (2011) also stated that 20 of the analyzed works show that US should include usability problems, and it was observed that prototyping is a great help factor in the integration of such methodologies. Such conclusions are also presented by (MEMMEL; GUNDELSWEILER; REITERER, 2007; SCHWARTZ, 2013; JURCA; HELLMANN; MAURER, 2014; GARCIA; SILVA; SILVEIRA, 2017). Plonka et al. (2014), through a case study, writes on possible ways to integrate UX and agile. Among the issues, difficult communication between UX practitioners and agile developers, as well as challenges on having design elaborated upfront development are highlighted. Similar problems are also presented by Silva et al. (2011) and Budwig, Jeong and Kelkar (2009).

Still regarding US, Lopes et al. (2017), conducting an experiment to find out how elements of HCI may be appropriate within User Stories. In addition, Lee and McCrickard

(2007) reflects on similar problems, pointing out that Scenarios are always revisited at the end of each iteration of the agile practices, also pointing out that the use of Scenarios as well as Mock-ups is important for Agile development processes improve their usability aspects, however, it is noted that the creation and management of such artifacts can take valuable time from methodologies aimed at having fast deliveries.

Regarding communication issues, in the work *How Artifacts Support and Impede Requirements Communication*, Liskin (2015), after conducting interviews with project development practitioners (with different roles), was able to map three main artifact types: containers, individual elements, and solution models. Further characteristics were also found, leading to a subdivision of the three main types earlier mentioned. Such division can be seen in Figure 5.

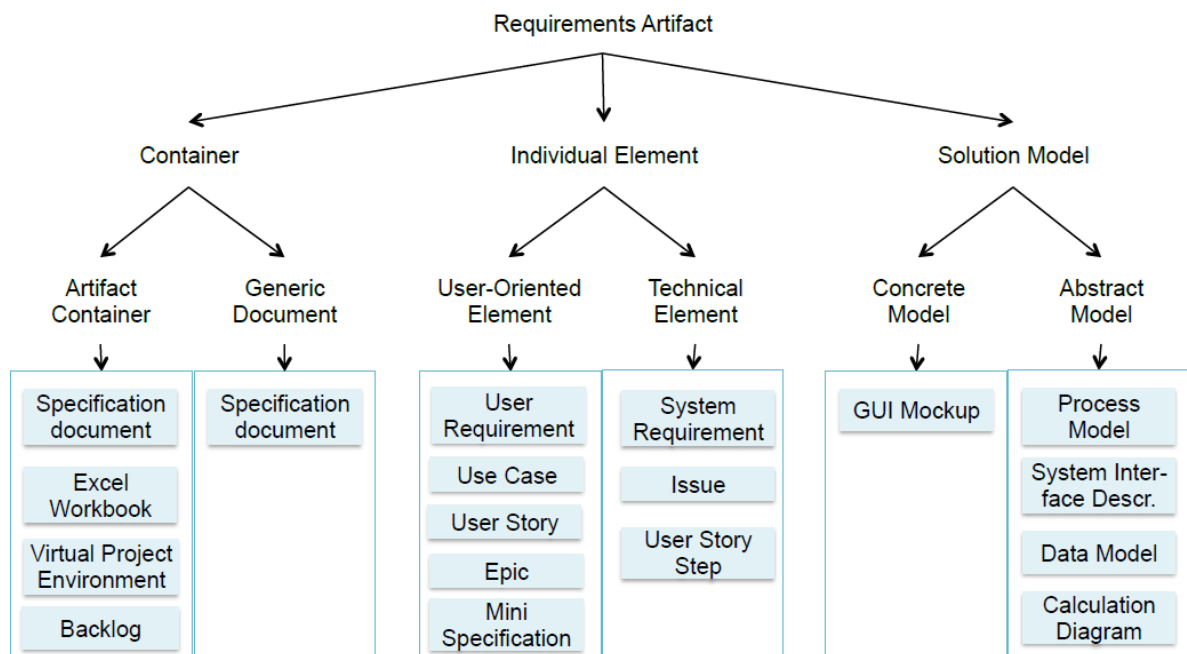


Figure 5: Classification of used artifacts

Source: Liskin (2015)

Among all the types presented by Liskin (2015), the *container* type, as well as the *individual element*, more specifically regarding US, are important to this master project.

According to Liskin (2015), *container* artifacts as the ones consist of other artifacts, holding the information together in one place. Requirement (specification) documents can be seen as containers, given that it holds different types of information, being displayed in a variety of ways, including different requirements and mixing the same with goals, policies, etc.

The *Individual elements* are divided into user-oriented or not, being such classification based on the user contribution to the element creation and assessment. USs can be

set in both sub-categories, given that it should be clear to the user, and contain steps and details to the developer, although the user may need a help from the developer, once that a US may not clearly relate to the actual business tasks in the user's daily work (LISKIN, 2015).

Still on communication issues, Garcia, Silva and Silveira (2017) displays in its Systematic Mapping, the most common artifacts used to facilitate the communication. Among the artifacts, it is possible to highlight the Prototype, as being the most common artifact that helps improving communication, leading to the conclusion that such artifact may contain valuable information within it. Other notable artifacts presented in the same work are User Story, Cards, Persona, Sketch, Scenario, Wireframe, Mockup, User Flow, Story Board, Use Case, among others. It is important to also mention that such artifacts are also present in the work of Silva et al. (2011).

Other results focused on artifacts and documentation, presented by Hess, Diebold and Seyff (2017), show that, in agile practices, have the level of detail of the artifacts (and documents, when present) is not high when compared to other practices. This may be due to the fact that the artifacts used in agile practices may not always contain all the necessary range of useful information such as links or quotes from other artifacts such as Personas or Mock-ups. These factors are part of the work of Schön et al. (2017), which proposes suggestions on how to work with different requirements, as well as in Soares et al. (2015), which identified that US may not be sufficient to support development activities such as software design and estimation, pointing out that one of the main problems with the requirements is the lack of information contained in the artifacts, which can be due to the fact that the agile practices do not prioritize both the documentation aiming to gain time. Such a conclusion is also corroborated by Liskin (2015), which concludes that managing many artifacts is really a problem, although the usage of different artifact is shown to be necessary. The work states that project managers need to keep in mind that developers need to have a more detailed view of the tasks (User Story, for example), but it also needs to have a view of the whole and where each piece (task) fits with the others (what would be provided by Container type artifacts, such as the software requirements documents).

Although the issues that the usage of US may have, it is still a very important and used artifact. Brhel et al. (2015), on a Literature Review, states that the employment of USs to describe features providing business value to the customer is popular, being mentioned in 21 (25.3%) of the publications analyzed in its review. In the same work, it is concluded that the prioritization of non-functional usability requirements in comparison to functional requirements remains challenging.

Although the usage of different artifacts is necessary, the use of multiple artifacts imposes challenges such as the dispersion of information or inconsistencies between artifacts,

which may not necessarily be grouped together into a single set (LISKIN, 2015).

When analysing how artifacts could facilitate communication in Agile User-Centered Design (AUCD) approach, Garcia, Silva and Silveira (2019) concludes that one of the main challenges faced while establishing the integration of agile and UCD is how to facilitate communication among the invariably distinct involved practitioners. As of Brhel et al. (2015), the study advocates the idea of artifact-mediated communication. Aiming to identify and understand the artifacts used to facilitate the communication between designers and developers in AUCD. Through a netnographic study in a globally-distributed online community of agile practitioners, Garcia, Silva and Silveira (2019) concludes that the community understanding is that the team must be cross-functional and the designer should be part of the team. Designers and developers communicate in different agile events using different artifacts as facilitators. The communication occurs throughout the entire AUCD flow, starting from discovery session, passing through all agile events including planning, iterative cycle, review, and backlog refinement engaging the whole team in design practices and UI specifications, and sharing design artifacts. Such conclusions reinforce the artifact as a mediator for the better agile-UX integration.

Kashfi, Nilsson and Feldt (2017) investigate the challenges software practitioners face on dealing with UX in software development (KASHFI; NILSSON; FELDT, 2017). Among the different challenges, the authors identified the need for mechanisms that improve the traceability between UX-related and other requirements. Schön et al. (2017) carried out a survey and presented the key challenges in agile Requirements Engineering (RE). One key challenge reported was how to manage the diversity of documentation that support agile teams in their work (SCHÖN et al., 2017). Liskin (2015) conducted interviews with software practitioners and as a result the author classifies the artifacts into different groups according to the artifacts role in a project. The findings showed that projects need to make usage of a whole variety of different artifacts, which carries the risk of inconsistencies or inefficiencies emerging from the dependencies between multiple artifacts (LISKIN, 2015).

## 2.7.2 Virtual Environments

Kassab (2014) points that Jira is the most used tool to support agile practices. The author conclusion comes from a survey that compares agile with waterfall RE (KASSAB, 2014). Hess, Diebold and Seyff (2017) reports that agile approaches have a strong focus on face-to-face communication instead of having documentation of RE in details (HESS; DIEBOLD; SEYFF, 2017). This emphasizes its early analysis, which showed that artifacts used in agile activities cover key requirements information, but its level of detail is often less than in traditional RE artifacts. The authors statement is based on the results they got from interviews and a survey with agile team members.

Considering that US is the most popular artifact used by agile practitioners (SCHÖN *et al.*, 2017), remaining as the central focus of development from the time it is created until the code is handed over (SHARP; ROBINSON, 2006), many works have proposed ways of how to write USs. In 2004, Cohn (2004) proposed a grammar as a pattern to the writing of US which is currently used by agile teams (COHN, 2004). Shortly, the grammar contains elements to picture out the users, their goals, and reasons to do something in the system. Choma, Zaina and Beraldo (2016) extends the proposal of Cohn (2004) in order to create the UserX story from which agile practitioners can add UX information in the US's body (CHOMA; ZAINA; BERALDO, 2016). The authors concluded that agile practitioners have difficulties in understanding how a single US can hold all the UX information they need, highlighting that Product Owners (POs) were having difficulties in understanding such usability concepts, and also they did not know how to incorporate UX issues in the product requirements.

Moreno and Yagüe (2012) proposed a mapping mechanism to help on incorporating usability requirements into USs (MORENO; YAGÜE, 2012). Although such proposal, issues are still to be answered, such as how to manage the size of USs with a relatively high number of usability mechanisms.

In a literature mapping about agile and UX practices, Garcia, Silva and Silveira (2017) concluded that virtual artifacts are mostly used as basis for development phases in agile practices (GARCIA; SILVA; SILVEIRA, 2017). The authors also reported that mock-ups and USs are often used in combination to support the agile developers work. In 2019, Garcia, Silva and Silveira (2019) carried out an investigation in an online agile community and the results show that mock-ups are adopted in combination with USs (GARCIA; SILVA; SILVEIRA, 2019).

Given the virtual artifacts, Silva *et al.* (2018), in an analysis of the current state of agile and UX Design integration, called Agile UXD, states that different teams, working in different contexts uses a variety of artifacts and techniques to create a shared understanding. Given such conclusion, where the usage of different tools is present, may also be a potential point to difficult integrating UX and agile.

For virtual artifacts, in a work regarding remote working, Deshpande *et al.* (2016) concludes that project tracking system like Jira are effective for both team members working in the office or remotely, being virtual artifacts and their supporting tools key information hubs for all team members. As described by Deshpande *et al.* (2016), information hubs can be taken as central focuses where information flows meet and decisions are made. Specially for the remote worker, these virtual artifacts dominate and shape their situation awareness (i.e. how people are kept informed of what is going on, through what they can see, hear or from what is accessible to them), being the core of their horizon of observation, which is what an individual can see or hear, influencing its situation awareness (DESHPANDE *et*

al., 2016).

In a study regarding requirement traceability in virtual environments, [Shukla, Auriol and Baron \(2011\)](#) highlights that the management of artifacts relationship is a continuous activity, involving people of various levels, to participate continuously and maintaining a perfect communication channel among them for avoiding any information lapse. As the requirements are continuously evolving through the life of a project, requirements are added, removed or modified. The linkage between these changing requirements needs to be maintained ([SHUKLA; AURIOL; BARON, 2011](#)). In a similar topic, [Lee, Guadagno and Jia \(2003\)](#) concludes that RE attempts to communicate the ideas and needs of the product's stakeholders to the engineers and developers who ultimately build the product. Communication and interaction lies at the heart of agile practices, however, as projects become increasingly larger and distributed, maintaining effective and up to date communication becomes error prone, work intensive and ultimately unmanageable. Requirements traceability refers to the ability to describe and follow the life of a requirement. In large projects, management of the links becomes burdensome, and accuracy depends on the frequency of updates. If not maintained correctly, links from one artifact to another may be incorrect, as requirements or other requirements documents are disposed of. The traceability is therefore rarely end-to end, as the time and effort required is more than the perceived worth ([LEE; GUADAGNO; JIA, 2003](#)).

### 2.7.3 Navigational Distance

[Bjarnason et al. \(2016\)](#), presented the Theory of Distances, which aims to categorize the difference of position or level among the stakeholders, artifacts, or activities involved into the development of software projects. The distances are related to the effort required to accomplish the tasks that constitute the development of the system.

They are classified into eight sub-categories, being one of them the navigational distance. Navigational distance describes how distant the artifacts are one each other regarding their positions ([BJARNASON et al., 2016](#)). In a virtual tool, such distance can be taken as the length of the path to navigate between the artifacts. [Bjarnason and Sharp \(2017\)](#) describe such navigation as being the length calculated by the number of links that the individuals have to access to reach the information.

### 2.7.4 Conclusions and Related Work Comparison

After the analysis of the presented literature review, it was concluded that the works show the advantages and disadvantages of the usage of several artifacts as well as the use of agile practices and UX techniques, showing how they could be incorporated in one single methodology. However, there is a wide range of artifacts available to be used

during the project life cycle, and such artifacts are not always correctly used together. Although the usage of US helps developers to understand user's needs, on the technical perspective, the information contained in such artifacts is not always sufficient, and other artifacts, which complement the necessary information (GARCIA; SILVA; SILVEIRA, 2017), are not always properly managed or have a clear form of communication for all those involved. The UX aspects are stated as recurring being overlooked in agile practices (GARRETT, 2010). But the studies do not cover how the UX information can be related to the USs.

Among issues highlighted in literature, loss of big picture (SILVA et al., 2011; GARCIA; SILVA; SILVEIRA, 2017), information traceability (LEE; GUADAGNO; JIA, 2003; KASHFI; NILSSON; FELDT, 2017), usage of UX elements into agile practices (ROGERS; SHARP; PREECE, 2019; INAYAT et al., 2015), and information dispersion (DESHPANDE et al., 2016; SHARP; ROBINSON, 2006) have been focused in this master project. Such focus is due to the fact that these issues can be related to the navigational distances (BJARNASON et al., 2016; BJARNASON; SHARP, 2017) into virtual environments, which is the topic of this master project.

Tables 1 and 2 show a comparison of the main related works here presented. Notice that the main subjects focused on the literature review were Agile, UX, US, Virtual Environments and Navigational Distance. Table 1 presents the summary of the main related works found presented in this chapter. Table 2 compares such works, considering the ids outlined in Table 1, with the topics aforementioned. Moreover, it compares such works with this master project.

Despite studies have presented different ways on integrating UX into agile practices, validating how communication and artifact usage may impact in such endeavor (BRHEL et al., 2015; SILVA et al., 2011; GARCIA; SILVA; SILVEIRA, 2017); the studies do not focus on how the UX elements can be found with agile artifacts. Being the US one of the most used artifacts in agile practices (GARCIA; SILVA; SILVEIRA, 2017), verifying how the UX elements relate to it is a gap still to be covered in the literature. Moreover, given the importance of virtual environments, specially to distributed teams, a practice commonly applied into agile teams (SHARP; ROBINSON, 2006; DESHPANDE et al., 2016); having in mind possible issues of navigational distance (BJARNASON; SHARP, 2017), the impact on relating UX information with USs is still to be further analysed, specially in virtual environments.

This study aims to focus on such gaps, focusing on how UX elements can be related to US in agile virtual environments, and what navigational distances of such relations are found out. Moreover, to propose templates on how to better create such relation in a way that the navigational distance in the virtual environment remains within a manageable limit.



Id	Title	Reference
R.W.1	Exploring principles of user-centered agile software development: A literature review	<a href="#">Brhel et al. (2015)</a>
R.W.2	Artifacts for Agile User-Centered Design: A Systematic Mapping	<a href="#">Garcia, Silva and Silveira (2017)</a>
R.W.3	UserX story: incorporating UX aspects into user stories elaboration	<a href="#">Choma, Zaina and Beraldo (2016)</a>
R.W.4	User-centered design and agile methods: a systematic review	<a href="#">Silva et al. (2011)</a>
R.W.5	Towards Requirements Communication and Documentation Guidelines for Agile Teams	<a href="#">Hess, Diebold and Seyff (2017)</a>
R.W.6	The role of distances in requirements communication: a case study	<a href="#">Bjarnason and Sharp (2017)</a>
R.W.7	How artifacts support and impede requirements communication	<a href="#">Liskin (2015)</a>
R.W.8	Integrating Agile and user-centered design: a systematic mapping and review of evaluation and validation studies of Agile-UX	<a href="#">Jurca, Hellmann and Maurer (2014)</a>
R.W.9	Key challenges in agile requirements engineering	<a href="#">Schön et al. (2017)</a>
R.W.10	Adding human interaction aspects in the writing of User Stories: a perspective of software developers	<a href="#">Lopes et al. (2017)</a>
R.W.11	Agile user stories enriched with usability	<a href="#">Moreno and Yagüe (2012)</a>
R.W.12	When user experience met agile: a case study	<a href="#">Budwig, Jeong and Kelkar (2009)</a>
R.W.13	Elements of user experience, the: user-centered design for the web and beyond	<a href="#">Garrett (2010)</a>

Table 1: Main related works.

Ref. ID	UX	Agile	US	Requi- rements Eng.	Arti- facts	Naviga- tional Distances	Contribution
R.W.1	x	x		x			Investigation on UCD usage along with agile software development
R.W.2	x	x	x		x		Systematic Review on which artifacts can be used in Agile UCD approaches.
R.W.3	x		x	x			How to include UX information in the US writing process.
R.W.4	x	x					Systematic Review on UCD and agile practices. Discusses which artifacts are used to support collaboration among designers and developers.
R.W.5	x	x		x	x		Guidelines for agile team on how to have better requirements communication and documentation.
R.W.6					x	x	Reports how distances may impact the project development, increasing the awareness of the same.
R.W.7		x		x	x		Conducted interviews in order to understand how artifacts help on software development on regarding information communication. Proposes artifacts classification.
R.W.8	x	x					Systematic Mapping on integrating Agile and UCD. Identified possible gaps in existing literature in order to enable future work to be done on such.
R.W.9		x		x			Identified 6 key challenges on Agile requirements engineering.
R.W.10	x		x	x			How to add HCI aspects into US given the developer's perspective.
R.W.11	x		x	x			How to add usability aspects into US.
R.W.12	x	x		x			Case Study on UX-Agile integration. describes challenges and recommend practices aiming to help future work on the subject.
R.W.13	x			x			How UX information can be present in software life-cycle phases. Establishes the five elements of UX framework.
This Master Project	x	x	x	x	x	x	Analysis on how the UX information related to USs. How such information is dispersed in a virtual environment. Recommendations to decrease the navigational distance.

Table 2: Related Work Comparison.

## 3 Case Study

Considering the background presented in Chapter 2, the following RQs have been outlined: RQ1 - *How are UX information and USs connected into software virtual environments?* and RQ2 - *What are the navigational distances found to access UX information from USs into software virtual environments?*, a qualitative case study has been done to answer the RQs.

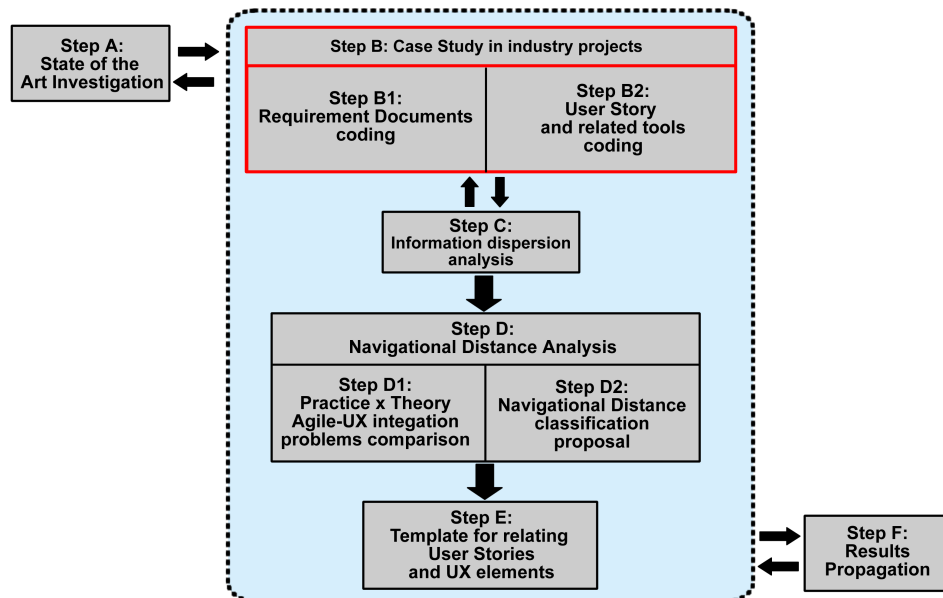


Figure 6: Case Study

Source: Author

This chapter presents such case study, on the following structure: i) We present a context on the settings that have been used in the case study. ii) The analysis of the case study is presented, being divided into four steps that are summarized by the uncovering artifacts process, the coding on software requirement documents, the coding on USs that were written based in such documents, and the results. iii) We end the chapter exposing the conclusions on the same.

### 3.1 Context settings

The case study was conducted in a software global organization of financial domain. It is present in twelve countries, having more than five thousand employees. The study concentrated on projects developed in Brazil. Despite the teams in the company adopted agile practices, not all the agile principles could be fully applied. The nature of the company

(i.e. financial sector) demanded controlled processes. Therefore the projects, and the teams, can be categorized as agile in non-agile environments (GREGORY et al., 2016). In such category, the agile teams apply agile practices, however, they use some structures and procedures closer to the traditional development process. In particular, they have more rigor in RE issues than the ones following agile practices.

In this organization, the writing of USs was done based on software requirement documents. Such documents were structure in sections, called Use Cases. The Use Cases work as input for the US writing, given that the Use Cases provide a set of co-related information in a big scope, later broken down into several US during the writing process. The UX information related to the project was collected in different ways. In some cases, artifacts (e.g. mockups) are created and embedded in the requirement documents. In other scenarios, different platforms are used for UX information storage, being such platforms linked to USs through hyperlinks. All the process of requirement specification until the writing of USs often follows the same steps. Briefly, we will describe these steps below.

First, the requirements are raised by the interaction of the Project Owner (PO) with stakeholders (end-users, managers, etc.). Considering the requirements elicited the PO and a requirement team are the responsible for elaborating the requirement document and for delivering to the leader of project. Taking this document in hands, the leader of the project has a conversation with the development team to decide how the new requirements can fit into a team work plane. The POs and the requirement team also answer the doubts that the development team can have about the document.

Subsequently, during a planning meeting, the leader of the project and the development team have a discussion based on the requirement document and then defined general USs that usually are known as epics. An Epic is a large User Story that cannot be delivered as defined within a single iteration or is large enough that it can be split into smaller USs (COHN, 2004). The USs are created in a "free" writing way, which means that they do not follow a pattern for its structure. The Epics are stored in Jira virtual tool. At the end of the planning meeting all the Epics are broken into minor USs that are linked to the Epics. Before the developers start to work with a given US, the leader of the project can make modifications or introduce additional information on the minor USs with the aim of clarifying some doubts that raised during the planning meeting.

Frequently, the USs are linked to other complementary documents that give information about UX or functional requirements. These complementary information is stored in different virtual repositories. UX information is found in extra documents stored into Confluence or SharePoint. Although the case study performed on requirement documents and in USs was based in the tools here mentioned (Jira, Confluence and SharePoint), the conclusions, recommendations and templates later presented in this master project didn't consider any particular characteristic of any tool. The analysis on the tools only

considered the UX information dispersion and navigational distances found in them in order to have insights on how UX information can be dispersed in virtual environments. We therefore believe that both the case study conduction and the results achieved from it can be applied in other tools and scenarios, although changes may need to be done in order to better fit in the particularities of each tool.

## 3.2 Analysis approach

To answer the RQs, we conducted a case study, which aims to investigate the contemporary phenomena in their context (RUNESON; HÖST, 2009). Figure 7 illustrates the four steps we followed in the case study analysis.

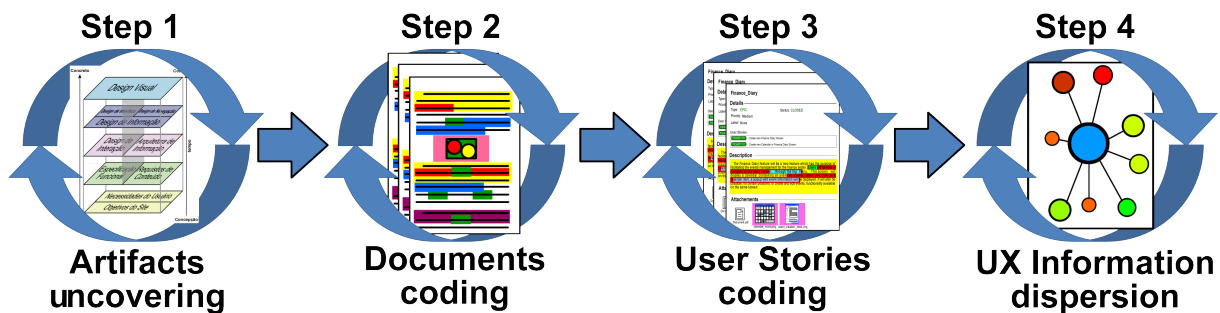


Figure 7: Steps of the analysis

Source: Author

**Step 1 - Artifacts Uncovering:** Manual step in which a total of 13 requirement documents were analyzed. Each document was structured in parts called use cases. The 13 requirement documents contained a total of 68 use cases. This analysis aimed to verify which use cases contained UX elements in it. Given the focus of this project, requirement documents that did not contain UX elements, would not generate USs with such elements in its content. Such documents could, therefore, be excluded from the further analysis described below, not being used in this case study.

**Step 2 - Documents coding:** Conducted in the 13 documents selected in the step 1 (for information about coding process see section 2). The coding process was performed in three steps. First, we applied the *closed coding* approach supported by Garrett's framework. We labeled a chunk of a document when we found out evidence of UX information. The codes assigned reflected the elements of UX of Garrett. After, we performed an *open coding* approach. In this step, we explored all documents creating new labels for UX information. These new labels represented UX information that was not explicitly related to Garrett's elements of UX. For instance, we assigned the label end-users when we found out details

about the audience. Finally, we performed a double check in all the documents to search for inconsistencies.

**Step 3 - User Stories coding:** We examined the USs following the same procedure performed in step 2. Taking into account that the requirement documents provided the information to the writing of the USs, we considered that we could find the same UX information embedded in the USs or related to them. In this company the USs (i.e. Epics and minor USs) were stored only in virtual tool (i.e. Jira). We carried out the coding process having the USs as our start point. During the coding of one US, we also examined whether that US had connections to other virtual tools (e.g. Confluence pages and artifacts stored in SharePoint).

**Step 4 - UX Information dispersion:** This step considered the step 3 outcomes, from which we noticed that most of UX information was spread in different virtual tools. To examine in-depth how the UX information was or not hold into the USs, we explored in which way the information was connected in the virtual environment. Such connections could be found through hyperlinks, attachment, mentions, and so on.

Although the documents and the US differ in each project, the process of creating them can be described as: 1) Requirements are raised by PO and stakeholders (final users team members, especially managers). 2) Requirement documents are written by a specific team (not the development one) along with Project Owner. 3) Requirement documents are delivered to development Project leader. 4) Once decided when the new requirements are to be delivered, the development team, whenever possible, reads the new requirements. 5) The development team has a planning meeting to overview the document and to elaborate the main User Stories, which are stored in Jira virtual tool. Questions are raised and sent to PO. At the end of meeting US are broken into minor US (i.e. a two round processes on US writing). 7) Quality Assurance (QA) members will add QA information over the next days, when each US is analyzed by them. 8) Project Leader (or PO itself) adds additional information to US regarding questions raised during the meeting. 9) Developers start to develop a US and add new information if the developer decides it is worthy to.

Despite each project has its particularities, the above described process is almost fully done by the three studied teams.

The USs, although not following a writing pattern, have a similar structure among projects, where they usually contain the same set of information required for the US content understanding: A text with a brief description as a subject; some acceptance criteria listed as bullet points and some relevant information described in a description section. However, each team used different details and artifacts and created a diversity of relationship among them. Besides Jira, Confluence supported the teams in the sharing of information and SharePoint in the management of the documents.

It is important to reinforce that, although the present study has been done over the mentioned tools, the results here presented were not derived from specific conclusions based in the particularities of each tool. The only factor that was considered is that the artifacts were stored in a virtual environment, instead of a real one. Also, given data confidentiality issues, no real data will be presented in this study. Rather than present real information from the documents analyzed, we will take fictitious data to illustrate our study. Documents and artifacts representing the analysis done in this study are available in Appendixes A to F.

Given the usage of Coding technique in this master project, it is important to explain how the same works. Below a brief explanation of the coding technique, including what is called *closed* and *open* coding, is presented. In this project, one of the steps consists in the document (text) analysis, in order to verify which artifacts are (explicitly or implicitly) used in the same. Such analysis will be done through a process called *coding*, which consists in creating a way to mark (code) the text making a parallel with the coded information and another (labeled) type of information. This process can be done by the "Coding" usage, which is described by Bohm (2004) as "*the deciphering or interpretation of data and includes the naming of concepts and also explaining and discussing them in more detail*".

The Coding technique is part of the *Ground Theory*, which, according to Bohm (2004) is a *Kunstlehre* (art), which means that its procedure cannot be learned in the form of prescriptions. The result of coding, will be a list of terms, as well as its explanations. For this study, at least two rounds of coding will be done, by using the "*closed coding*" and "*open coding*" techniques.

The process of *closed coding* was done by using information previously obtained in Garrett's Framework (GARRETT, 2010) analysis. In this coding, artifacts and UX elements mentioned in Garrett (2010) work were used as labels for the coding done in the documents analysed.

The process of *open coding* was also used in this project. Such process can be considered as an analysis through which the data (text) is broken down, each word or line is analysed and coded, after the first round, larger pieces or blocks of text are analysed, and coded as well. In the end of such data analysis, a succession of concepts is developed, being ultimately used to build blocks for a model (BOHM, 2004).

The coding technique allows someone to keep distance from existing theories, allowing the theory to grow out from the data itself. Given such characteristic, Bohm (2004) states the limitations of such technique to be dependent on the investigator's creativity, which makes it learnability difficult.

### 3.2.1 Artifacts Uncovering

For this first step, a manual process of reading requirement documents was performed, aiming to filter which requirement documents could be used for this case study. First, we examined a set of project documents seeking for those that contained UX-related information.

All the documents were structured in sections, being the main information to the development team located into sections named Use Cases<sup>1</sup>. After exploring the documents we concluded that apart from the use case sections, the others contained database structures and other information that was not relevant for our purpose (i.e. these did not have relation to UX information), therefore these sections were removed from further analysis.

After filtering the documents we proceeded in the analysis by reading carefully each document, searching for any UX information. A document was selected whether at least one UX element was found out. For instance, the use case made a relationship with some part of a mockup. We also found out that the project document had links to other documents or UX artifacts.

This case study step had its analysis based in Garrett's framework (see section 2.5), in which the UX elements mentioned in such framework were considered in the analysis. The results of this first analysis, which aimed to filter the documents that would be used for the further analysis steps, are presented next:

- For project 1, 30 use cases were analyzed, distributed in 6 different documents. A total of 21 use cases contained aspects related to interface (UX) aspects;
- For project 2, 18 use cases were analyzed, distributed in 4 different documents. A total of 8 use cases contained aspects related to interface (UX) aspects;
- For project 3, 20 use cases were analyzed, distributed in 3 different documents. A total of 6 use cases contained aspects related to interface (UX) aspects.

A total of 13 requirement documents, containing a total of 68 Use Cases were analysed. From these, 35 Use Cases contained aspects related to UX elements.

### 3.2.2 Documents Coding

In this case study, three projects had their software requirements documents analyzed. The study was done with real documents but given data confidentiality, no real data will be presented in this document. The documents here presented are versions

---

<sup>1</sup> Use Case is a generalized description of a set of interactions between the system and one or more actors, in which an actor is a user or another system (COHN, 2004)



developed by the proponent of this work, in which the information necessary for the study is presented in a "generic" format, omitting any details regarding the company or its projects. Such documents will be referred to as "case study documents". The case study documents were written aiming to describe the content of the real documents analyzed, without presenting details that compromise the company. The case study documents can be found in Appendixes A, B and C, already containing the coding process.

The following sections were disregarded from the documents: Database Structure and/or Diagrams; Questions and Answers; Non-Functional Requirements; Future Improvements; List of Terms; Cover; Index and List (of Terms, Figures, Tables, etc.). These sections were removed given that they did not provide new insight into the analysis being made. Such sections did not contain relevant information for this study, and only contained information or artifacts that are not related to the user experience.

The purpose of this case study was to explore artifacts and UX elements that were present in the requirement documents. In a later step, this case study aims to verify which of these artifacts and UX elements were also found in User Stories. The result of this case study, therefore, is to verify the RQs defined in this master project. The RQ1 aims to understand "*How are UX information and USs connected into software virtual environments?*". Therefore the need to investigate how UX elements relate to US into the agile projects is required, considering the usage of virtual environments. The finding of UX elements into the software requirement documents will later be used to see whether such UX elements are also found in the US level, considering that the USs are written based on such documents.

With the use cases that contained UX aspects, resulted from case study step 1 (subsection 3.2.1) a second analysis was started, in which it was intended to find a pattern in the writing of the requirements. This analysis would lead to the creation of the use case documents, appendixes A, B and C, previously mentioned. The analysis was done in an ad-hoc manner, and resulted in the creation of the attached documents (Appendixes A, B and C), which aim to be representations of the real documents analyzed in this study, without presenting confidential information.

During the creation of the use case documents, the artifacts and UX elements that were present in the original documents were also analyzed. Such artifacts and UX elements also had "generic" versions created, which represent the real artifacts found in the document analysed, but without confidential data. The artifacts and UX elements created are available in the use case documents. The use case documents present the artifacts and elements of UX that were found in one or more documents within the same project. For example, in project 1, although 21 use cases have been analyzed, only some of them could contain a particular artifact, for example a Mock-Up. Although the Mock-Up artifact was not present in all use cases, given its recurrent usage, it was chosen to include it in the

generic version of the requirement documentation.

The coding process was done in two parts. The first is classified as *closed coding*, in which information previously obtained through the analysis of the work of [Garrett \(2010\)](#) were used in the coding. The second part was *open coding*, in which the documents were explored in order to obtain new codes that might not be related to those used during closed coding.

The first step of coding was based in the UX elements and artifacts mentioned in the work of [Garrett \(2010\)](#). A analysis of the Framework (see [Figure 4](#)) resulted in the finding of some artifacts/UX elements being mentioned in a specific plane from the framework. Such artifacts/UX elements, and its respective placement in the Framework, resulted in one column, named *Garrett Framework* in the coding result, later displayed in [Figure 12](#). Such column has the intention to be a common reference to the analysis explained in [section 3.2.4](#).

In a first analysis, each of the documents was analyzed in sequence, according to the project. From now on, the documents related to project 1, will be called Document 1, for project 2, Document 2, and for project 3, Document 3. The first analysis was done using closed coding. Starting from Document 1, the closed coding process, analyzing the artifacts and UX elements, presented by [Garrett \(2010\)](#) was performed. For each element found, the textual fragment, image, etc., was marked with a color that would later be used as a label, which represents an artifact or specific UX element. Each color, or label, was then used in new occurrences of the same artifact or UX element. The number of occurrences of a particular label in the same document is not part the scope of this study. As many occurrences of a given label happen within the same document, such label is only counted as one in the final analysis.

Once the analysis was performed in document 1, another analysis was carried out in document 2, and then in document 3.

After the end of first coding, a second coding process began. This second coding happened in two iterations. In this second coding, besides the refinement of the coding done at the first process (*closed coding*), the *open coding* technique was applied. This analysis was done in order to find possible new codes, which were based on the elements that had been found during the literature review (see [chapter 2](#)) but were not directly mentioned in the work of [Garrett \(2010\)](#).

In this second coding cycle, the coding was done focusing more in the textual elements. The textual elements could, sometimes, contain some code previously mapped elsewhere in the document, but for completeness purposes, any mentioning of a previously found codes should be mapped to the corresponding color. Such mapping eventually generated results as shown in the [Figure 8](#).

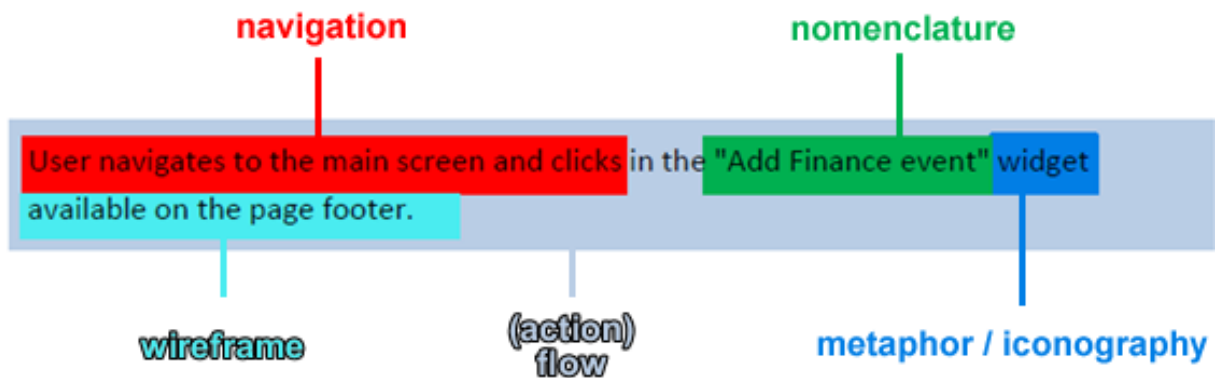


Figure 8: Coded text chunk

Source: Author

As seen in Figure 8, a small text chunk could contain different codes, in this example, five different codes. Although some of these codes are present in other parts of the document, in order to have a more complete coding, each appearance of the codes has been mapped. This step provided initial insights for the third iteration of coding, later described.

Once the process described above was done in document 1, the same process occurred for document 2 and document 3.

At the end of this analysis, all documents from the three projects were analyzed again, taking into account what had previously been mapped in the first coding iteration. The second iteration aimed to find coded elements only found in documents at the end of the first iteration, which had not been mapped in documents coded at the beginning of the same. For example, a code that was only discovered in document 3 would probably not be mapped in document 1, so this second iteration covers this possible gap.

At the end of the second iteration of coding, the mapping of the codes resulted in ones presented in Figure 9, which later became labels, used in section 3.2.4.

A third, and final, coding iteration was performed, aiming to verify if the mapped artifacts and UX element were used directly or indirectly. For this analysis, the coding technique was used again, but this time using a new coding methodology, based on the direct or indirect usage of the elements. The labels annotated with *D* indicate the direct presence of a given artifact or UX element in the documents. Labels annotated with *I* indicates that a given artifact or UX element were not present in the documents, although related information can be found in the same.

The Direct (*D*) and Indirect (*I*) coding intended to provide a second information for each of the codes previously found in the documents. The artifacts and UX elements

	scenario		gherkin scenario
	strategy / overview		quality assurance
	wireframe		security requirement
	navigation		use case diagram
	flow		as a user I want
	nomenclature	<b>D</b>	Direct, explicit (usage in document)
	persona	<b>I</b>	Indirect, implicitly (usage in document)
	mockup		
	typography		
	color palette		
	mataphor / iconography		
	structure / architecture		
	use case		
	user story		
	data type		
	pattern, consistence		
	error handling		

Figure 9: Coding process codes

Source: Author

that were explicitly (directly) used in the document, received the *D* tag. For example, if document 1 contained in its content a Mockup artifact attached; the Mockup coding (with the color corresponding to the label) was made in the document. In addition, for the codes of document 1, the Mockup code received the tag *D*, representing the direct, or explicit, usage of such artifact.

In contrast, the coding *I* was used when a particular artifact was not explicitly or directly used in a document. For example, in Document 1, information related to Users, such as access permissions, Role grouping, etc., could be commonly used. A brief description of the user was also provided in the text format. Such information could have been presented through a Person artifact, but such artifact itself was not found in the documentation. Given the presence of such information could have be presented in a Persona artifact, the usage of *I* tag, meaning indirect or implicit usage, was applied. Such tags therefore indicates that a particular artifact, or UX element, is not used in the document. However, information that could be used for the creation of such artifacts, or could have come from an existing artifact, is found available in the document. In most of the cases, the presence of such information was made through text, while no artifacts were used.

### 3.2.3 User Stories Coding

After coding the documents, USs were also analyzed, following the same coding process. This next coding was done to understand which artifacts or UX elements, found during the coding of the requirement documents, were transferred to the US level.

Taking into account that the requirement documents provided the information to the writing of the USs, we assumed that we could find the same UX information embedded in the USs or related to them. This assumption would be verified through the coding of USs. Later, an analysis of the UX information dispersion would also be done, in case some UX information was found outside the US.

In this company the USs (i.e. Epics and minor USs) were stored only in virtual tool (i.e. Jira). We carried out the coding process having the USs as our start point. During the coding of one US, we also examined whether that US had connections to other virtual tools (e.g. Confluence pages and artifacts stored in SharePoint). Figure 10 shows an example of coding in an US.

**Create new Calendar in Finance Diary Screen**

**Details**

Type: Story Status: CLOSED  
 Priority: Medium Fix Version: 2019.1  
 Label: None Epic Link: Finance\_Diary  
 Story Points: 5

Acceptance Criteria:

- Validate that calendar is displayed with current month when access the page.
- Existing events are pre loaded in the calendar according to the colors.
- Validate that Past events are gray
- Validate that Holiday type events are green
- Validate that Payment events are Blue
- When clicking in an event, validate the information in the pop-up

**Description**

The new Finance Diary screen should be displayed containing a calendar with the current month and all events loaded in it.

The user can navigate through the calendar months by clicking on the arrows.

When clicking on an event, a Pop-Up should appear showing the event information.

Calendar events should be colored according to the following rules:

- Gray: Events in the past
- Green: Holiday Events
- Blue: Events of type Payment

**CODING**

- information flow
- quality assurance
- color palette
- navigation
- structure / architecture
- mockup

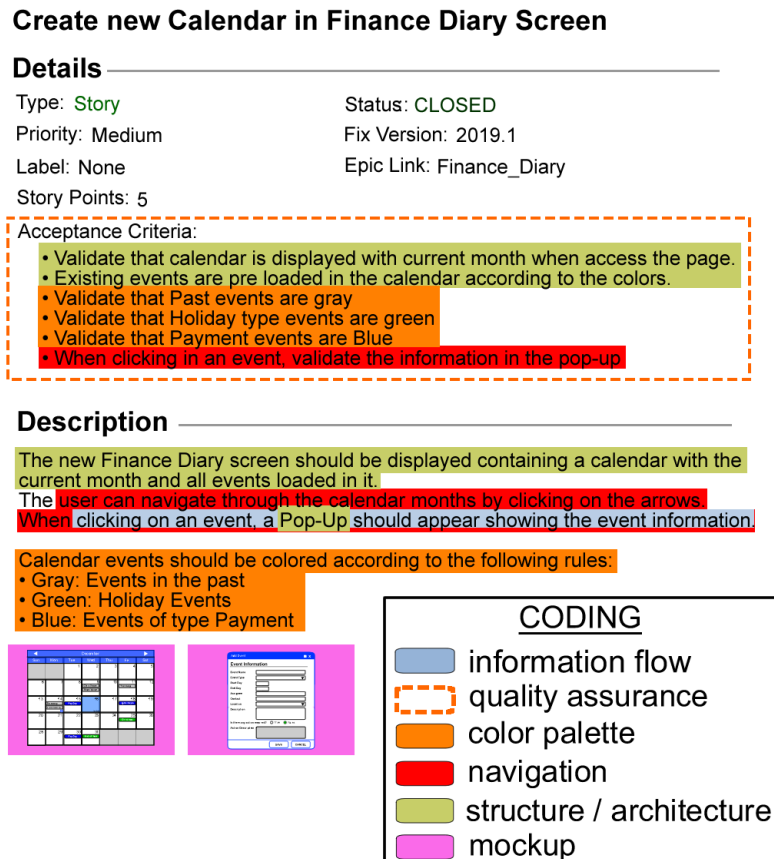


Figure 10: User Story coding sample

Source: Author

The analyzed USs were stored in Jira tool. The hierarchy used in the Jira is: Epic

> User Story > Task. All three hierarchy levels have been analyzed. For the task though, no information could be coded, since they were used only for development task division purposes, not for documentation purposes. An example of a Task is shown in Figure 11. For the Epics and USs analyzed, the coding process can be found in Appendix D, E and F.

## Create New Screen with edit operation

### Details

---

Type: **Task**

Status: **CLOSED**

Priority: **Medium**

Fix Version: **2019.1**

Label: **None**

Epic Link: **User**

Story Points: **5**

Acceptance Criteria:

### Description

---

UI Creation

Figure 11: Task example

Source: Author

Considering Step 3 outcomes, we explored the UX information dispersion (Step 4). We could notice that most of UX information was spread in different virtual tools. To examine in-depth how the UX information was or not hold into the USs, we explored in which way the information was connected the USs. Such connections could be found through hyperlinks, attachment, mentions, and so on.

### 3.2.4 Coding Results

The result of the coding process performed on documents is summarized in the mapping presented in Figure 12.

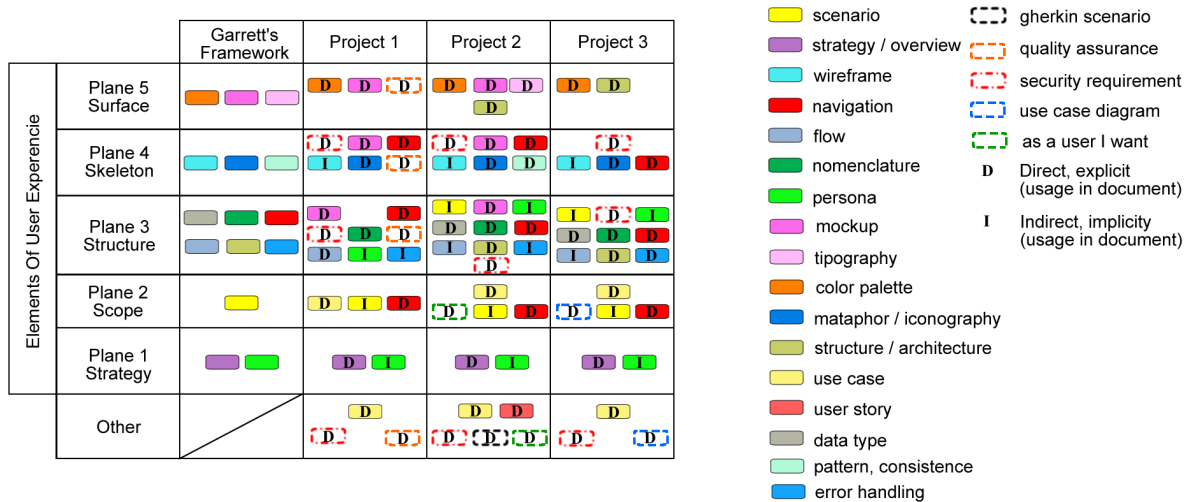


Figure 12: Document coding mapping

Source: Author

Figure 12 shows, in the *Garrett's Framework* column, the UX elements and artifacts mentioned in the work of Garrett (2010). In Figure 12, the labels, for *Garrett's Framework* column, are placed in the row corresponding to the Framework's plane (see section 2.5) in which such labels, representing UX elements or artifacts, are mentioned in the work of Garrett (2010). The last row of Figure 12 (row *Other*) highlights the labels that were found during the coding process but are not part of the Garrett's Framework (GARRETT, 2010).

In Figure 12, the *Project 1*, *Project 2*, and *Project 3* columns represent the labels found in the respective software requirement document, according to its project. At first, the labels were placed in the same plane of the ones they are mentioned by Garrett (2010). In the third iteration of the Coding process, given that some codes did not match UX elements or artifacts mentioned by Garrett (2010), such codes were placed in the *Other* row. After the end of the coding third iteration, it was noted the possibility of proposing the inclusion of such elements within the planes proposed by Garrett (2010).

An example of such is the Use Case, which is not part of any plane of the original framework. Once such artifact was found during the coding, and given previous knowledge obtained by the framework analysis (section 2.5), the idea for the placement of such artifact in the framework originated. Given such possibility, the suggestion of using the UC in Plane 2 (scope) was proposed. This proposal is based on the fact that, there is some relation between the purpose of the UC and any of the Planes proposed in Garrett's

Framework. Since the purpose of the UC is to state what will, and will not, be within the scope of a given development, it is possible to correlate it with the information contained in the Framework's Scope plane, that is the one responsible for the scope of the project information. Such analysis resulted in the UC label being placed in the *Plane 2 - Scope* in Figure 12.

This analysis was initially done in the *Other* row codes. The result of trying to include new elements in Garrett's Framework led to the idea of extending such analysis to the other codes found during the coding process. The other codes, placed in the corresponding Garrett's Framework Plane row, were then performed.

This new mapping was carried out based on the descriptions of Garrett's Framework planes (see section 2.5) as well as on the information contained in the artifacts and UX elements mapped in the documents. Some of the elements presented more than one information, which, in the context of the Framework Planes, could lead to its usage in different planes. For example, the Mockup artifact, present in the documents, can be considered a mid-level Mockup (representing, in a static image, a solution very close to the final result proposed). This artifact can be seen as a container artifact (LISKIN, 2015), grouping information such as color palette, label nomenclature, component positioning, and others. These information can be considered UX elements.

Considering such analysis, it is possible to state that there is the presence of UX elements within an artifact, in this case the Mockup. Still in this example, it is possible to see that these UX elements are mentioned in Garrett's framework, but in different planes. Therefore, although the Mockup is an artifact itself, located in a specific plane in the framework, we can state that it is also part, even partially, of other planes of the framework, in which the UX elements contained in the Mockup are part of.

Following the same analysis described above for the Mockup artifact, other analyses were performed, is summarized below.

The Mockup artifact is placed in Structure, Skeleton and Surface planes. This was done given that it contains nomenclature/labels, which are part of the Structure plane. Metaphors (icons usage) and components positioning (covering the purpose of the Wireframe artifact), which are part of the Skeleton plane, are also part of the Mockup. The Mockup also contains typography, color palette and general aesthetic information, corresponding to the Surface plane.

As for the structure/architecture artifacts are not artifacts, but in this study they are being considered as such, given that there were textual fragments in the documents analyzed that contained information exclusively on its aspects. This artifact is placed in Structure plane, according to the Garrett's Framework. It is also placed in Surface plane, since the information stored in them can refer to components usually placed in Mockups,



such as descriptions of how a certain information will be displayed on the screen. For example, a certain information should appear in a combo box component or in a Pop-up).

The Persona artifact is in Strategy plane given the original framework mentioning. It was also placed in Structure plane, since information of the users can be found in the descriptions of functionalities, which are part of Structure plane. Although the Persona artifact is not used in the documents, but information that constitutes such artifact could be found in the documents, the same was tagged as *I*.

Pattern and consistency is not an artifact itself, but is an important aspect to be considered during application development. It is part of the fourth plane in Garrett's framework. It may also be part of Structure and Surface planes, once that Mockups can be created already covering such aspects, specially mid-level Mockups, such as the ones used in the documents analyzed.

Regarding the Scenarios, they are mentioned in the Scope plane in Garrett's framework. Although the Scenarios artifact is not used in the documents, it is tagged as *I*, given an overview of the requirement scope could be found, which is similar to what a scenario would represent. The Scenarios coded in the documents show that they could contain navigation information. Navigation aspects are covered in the Structure plane of the Garrett's Framework, therefore the Scenarios are also placed in this plane in Figure 12.

In Project 3, the usage of the UC diagram can be seen. Although such artifact is not part of Garrett's framework, it could be used in Scope Plane, given that UC diagrams tend to display, in a image, the high-level scope of what will be developed, displaying user actions in a relational diagram. Therefore the UC diagram can be a visualization of the scope, information related to the scope of the development can be found. The UC diagram could also be used to illustrate what is stated in a scenario artifact, since that scenarios are usually done only in a writing manner.

The security requirements are also not mentioned in Garrett's framework, but they could be used in Structure plane, along with error handling information, which are mentioned in the original Framework. Depending on the software requirement, security requirements could also relate to user access. With such information, security requirements would relate to Skeleton and Surface planes, given that security access could relate to information visualization and application functionalities access.

Quality Assurance, which relates to the acceptance criteria, analyzed later in the US, is also not mentioned in Garrett's framework. Quality assurance aspects could be placed in Structure, Skeleton and Surface planes, given that: It can relate to the structure of the application, stating and assuring what a users may or may not access. Depending on the details present in the topics of quality assurance, aspects of navigation and positioning of components can be validated, relating to Skeleton plane. Quality Assurance can also

encompass user sentiment and validate aspects that are only covered by human interaction with the application, such as the response time that a given component takes to be loaded in the screen, or the color tones being used, etc. These aspects are at the highest level of the framework, therefore also relating quality assurance aspects to the Surface plane.

Apart from the labels placement in Figure 12, the first analysis resulted in some results that show the difference, in amount and placement, of the artifacts used in each project. The *DOC 2* is the one showing the greater amount of artifacts, although *DOC 1* and *DOC 3* also present a good amount of different artifacts. It is important to notice that, as per Figure 12, not all artifacts mapped are elements of UX. The artifacts that are not related to UX and are not cited in the work of Garrett (2010), see section 2.5, were placed in *Other* row in Figure 12.

Some common UX elements as well as overall artifacts can be found in all three documents analysed. From these, some remarks can be highlighted: Although some artifacts are mentioned in literature review as being commonly used, some of them could not be explicitly seen in any of the documents analysed. Such artifacts/UX elements include Scenarios, Personas and Wireframe (see chapter 2). Despite such artifacts are not explicitly present in the documents, as can be seen in Figure 12, they have been mapped as implicit (*I*), given that written text chunks were found, containing information that could relate to the artifact's information. Although some artifacts mentioned in Literature Review were not found in the documents, others could be found. From the same studies presented in Literature Review, it can be checked that, the Mockup artifact is mentioned as one of the most used elements, and the same could be found in two of the three documents analysed.

From this first analysis, an insight was also noted, which relates to the Agile practices. The planes proposed by Garrett (2010), although applicable in a agile project, may not have a very large distinction between themselves, in an Agile scenario. Specially considering temporal distances from the actions taken among the tasks executed in each plane, which occur in difference project life-cycle moments. Given that Agile practices tends to have an overview of the whole development to be done in each one of the small delivery cycles that constitute the life cycle of the project, the planes of Garrett's Framework, in an agile scenario, would also need to be considered as a whole.

This first analysis already showed the difference of artifacts usage among different projects/documents. It also didn't correspond to some data presented in previous studies mentioned in Literature Review (see chapter 2). Insights from the the data exposed in the work of Garrett (2010), as well as the artifacts and UX elements presented in the same, have also been be shown.

The second analysis was then done in the other artifacts used by each project, taking special look in the User Stories. This coding was done using the same methods applied for the documents, and can be seen in Appendix D, E and F.

The code mapping result for the US was summarized as presented in Figure 13, which was created following the same principles of Figure 12, except by the code placement in different rows when relating them to Garrett’s Framework. Although in documents mapping the findings have been placed in more than one plane from Garrett’s framework, for this mapping the same have not been done, as its intent was not to confront the artifacts or UX elements information with the work of Garrett (2010). Instead the purpose of this mapping was to discover which elements were found outside the documentation, in order to understand the information dispersion among the artifacts.

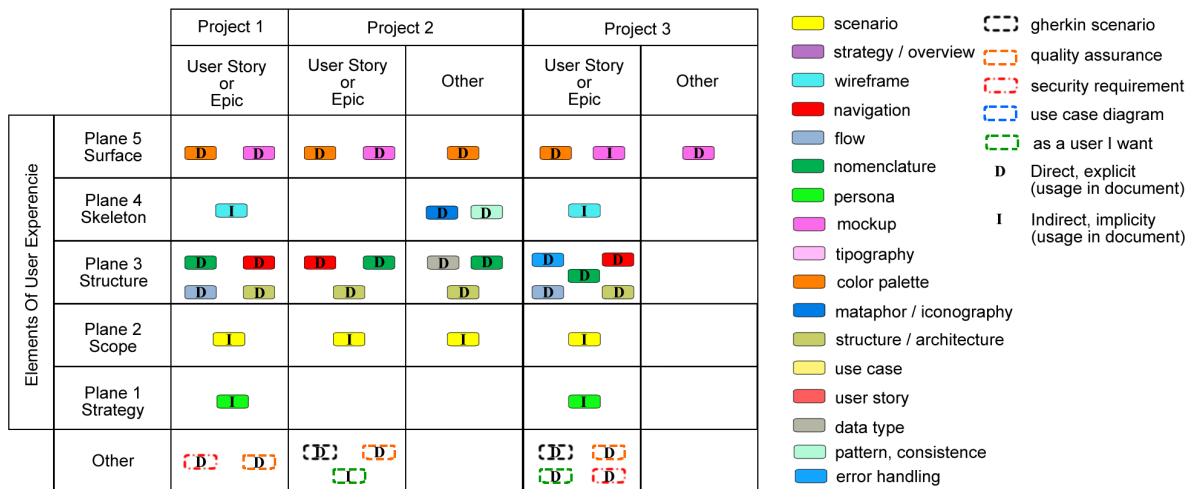


Figure 13: User Story coding mapping

Source: Author

The first thing noticed after the mapping was done is that not all previous mappings could be found outside the documents itself. From the document mapping, the coding number was 22, but on this coding the result was 17. The following artifacts could not be found during the coding process: Strategy/Overview, Typography, UC diagram, US and UC.

From these, US and UC coding missing don’t make negative impact to the analysis. This is because the US and UC are artifacts by themselves, in which the coding is being done. Given such scenario, it would not be possible to code a US within a US, or a UC from a UC. The UX element typography missing also don’t have impacts to the analysis, once that such information can be found in Mockups, without the need of explicitly mentioning the typography information in the US that already contains a Mockup in it. Therefore the artifact that could not be mapped and that may have some impact in this analysis are the Strategy/Overview and the UC diagram.

The coding process executed in US resulted in Figure 12, but further analysis are still to be done.

In an overview analysis, considering all codes for all documents, no further results, apart from the ones stated above, could be found. A further analysis, considering the *Direct* and *Indirect* codes of the mapping was then done. In this analysis, a matching, between the document codes and the US codes, for each project, was targeted. Each coding found in the documents was compared to the ones found in US level, for the respective project. The UC and US codes were not considered in this second analysis, given that such artifacts are now being analyzed. The differences in the matching are summarized below. In the following analysis, the "*first mapping*" refers to the analysis done exclusively in the documents (Figure 12), while the "*second mapping*" refers to the mapping done in the US level (Figure 13). The *D* and *I* coding mentioned in the below analysis correspond to the findings from the first mapping.

For Project 1, from the mapped elements found in the first mapping, the following could not be found in the second mapping: Strategy/overview (D); metaphor (D) and error handling (I).

For Project 2, the following were not found in the second mapping: Strategy/overview (D); typography (D) and Wireframe (I) (although these can be found within Mockup); security requirements (D); persona (I); error handling (I) and information flow (I). In the Project 2, the datatype (D); metaphor (D) and pattern/consistence (D) could only be found in Confluence pages, not being present at US level.

For Project 3, although not having Mockups present in the documentation, they have been found in other places (Confluence page, and in US level there is a link to the confluence page). Despite not using the "*as a user I want*" writing method in the document, the same was found in user stories. Apart from it, the following could not be found in the second mapping: Strategy/overview (D); data type (D); Use case Diagram (D) and metaphor (D).

Such results are also presented in Figures 12 and 13 in for of Venn Diagrams. Such presentation was created in order to understand which artifacts and UX elements were, or not, passed from the original requirement documentation to the US level.

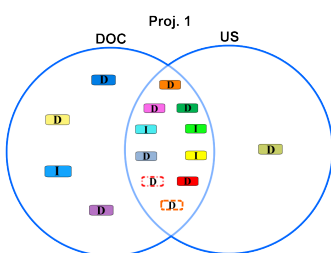


Figure 14: Venn Diagram - Project 1

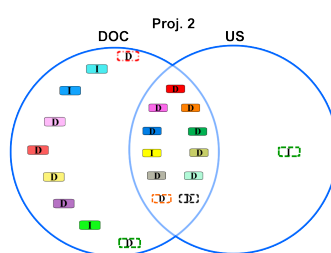


Figure 15: Venn Diagram - Project 2

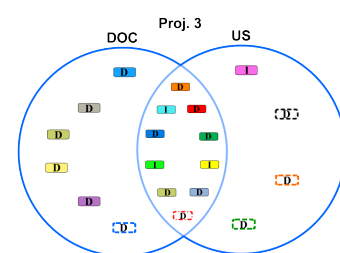


Figure 16: Venn Diagram - Project 3

Source: Author

As per Venn diagram on 15, it can be seen that project 2 contains more codes when compared if projects 1 and 3. It is also possible to highlight the artifacts that were only found in requirement documentation or only in US, previously named.

Synthesizing the codes previously displayed in Venn diagrams on Figures 14 to 16, the Venn diagram on Figure 17 displays a mix of all three projects codes. In such diagram, if at least one of the three projects showed a given code, the same was mapped into the diagram. Given this premise, if one of the projects showed a code being found in both requirement documents and US, the same would be added in the diagram intersection, despite in other projects such result may not be true. This was done in order to find which codes we could not found as mapped from requirement documents to US level in all three projects.

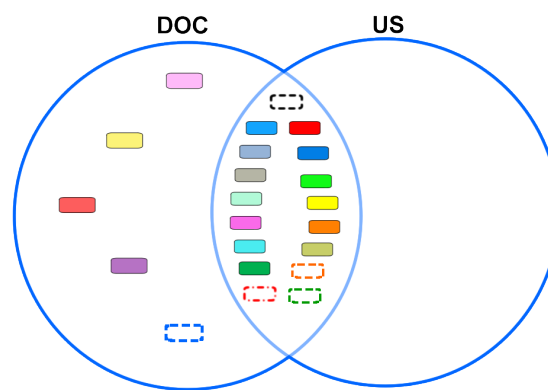


Figure 17: Coding Grouping - Venn Diagrams

Source: Author

As per Figure 17, it can be seen that no code were found existing only in the US level. On the other hand, five codes remained as being only found in the requirement documents. Such codes are, according to coding label presented in Figure 9: Typography, Use Case, User Story, Strategy/Overview and Use Case diagram.

For such codes, the following conclusions could be drawn:

1) The Use Cases were only used in all projects as an structure for the requirement documents, and despite being a possible representation of an Epic in the Jira tool, the UC code missing in the US level, do not cause loss of information.

2) The US coding was not found in the US level because it would require the USs created in the virtual environment (i.e. Jira) to be a copy of the User Story description in the requirement documents. Given the US creation process described in Chapter 3, the missing of such code in the US level also do not cause loss of information.

3) Use Case diagrams may not be required in a US level, if the information the

diagram presents are found in other way, for instance, if there is a US that describes the type of access a given screen will have, permission which is also expressed by the UC diagram. This coding exists in the US level but in other (written) format, and therefore do not cause loss of information.

4) Strategy/Overview, which comprehends a written description of the requirement documents purpose, could not be found in the US level, but in further analysis it was found in other locations (i.e. Epic or Confluence page). This indicates the presence of a navigational distance, topic later covered in this study. Given the information was present in the virtual environment, even though it was not in the US, it can be concluded that there was not loss of information for this code. 5) Typography could not be found outside the requirement documents, being therefore a loss of information. Given the projects contain standards on the development level to standardize the typography throughout the application, and that the typography present in the documents may be miss-aligned with such standards, and given the code, even though found in the requirement documents, was not a highlighted requirement, it can be concluded that the loss of information, for this particular code, would not cause a disruption in the project.

The next analysis will investigate that, although some artifacts and UX information initially found in the requirement documents were also found in the USs, not all US contains all the elements coded in the documents. Considering the agile practices, the need to break the work into small pieces, not requiring all the data do be stored in a single US, makes the US a representation of a small development to be done (COHN, 2004). Therefore, the next analysis focused on exploring how the information is spread across multiple USs and how its related information is dispersed in different virtual environments (i.e. Jira, as well as other platforms, such as Confluence). This analysis aimed to understand and evaluate the navigational distance of the information present in the different artifacts and UX elements already coded.

### 3.2.5 UX Information Dispersion

As seen in Figure 18, this section covers the analysis performed after the coding performed in requirement documents and User Stories. Despite still being part of the case study, in Figure 18, the following UX information dispersion analysis is separated from the others given its importance to the case study. In this analysis, the insights for the later exposed recommendations and templates that aim to decrease the navigational distances were found.

After the coding was performed (Section 3.2.4), the conclusions showed that UX elements could be found in all three projects (Figures 12, 13 and 17). During such coding process though, it was noticed that some of the USs analysed, which are stored in Management tool (Jira), contained external links to other applications, being such

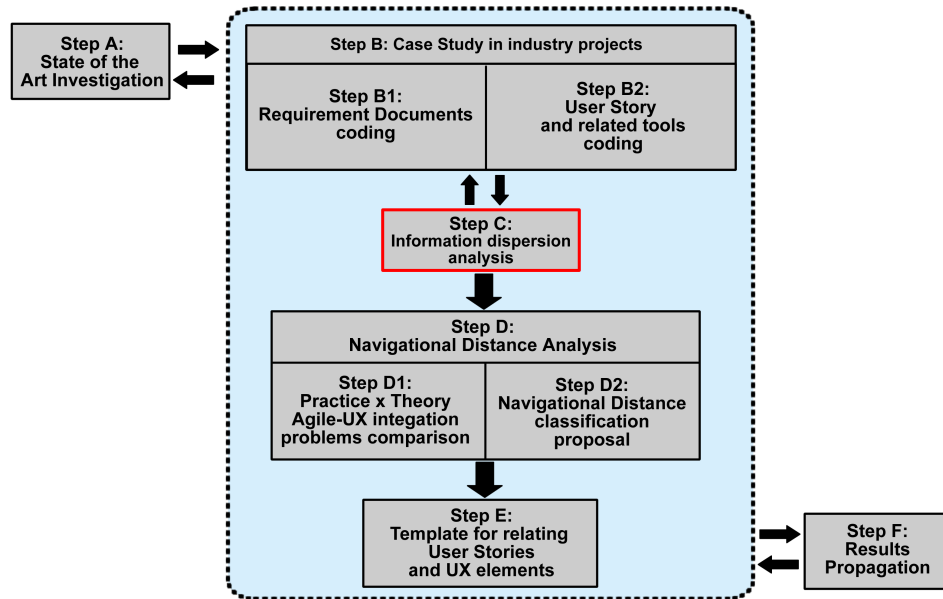


Figure 18: Case Study

Source: Author

Repository tools (Confluence and SharePoint).

The presence of UX artifacts, early found in requirements documentation, was seen in such virtual environments, although each project uses them differently. An overall placement and relationship among all such virtual environments can be seen in Appendix G. The usage of the previously mentioned tools per project was done as follows:

Project 1 only uses Jira to keep track of its information. Documents were attached to Epics, while other specific information were directly stored in US level, being it through attachment or in the text of its body.

Project 2 uses Jira and Confluence, having a special page for UX-related information (mockups, icons, etc.). Given both Jira and Confluence are from the same company, Atlassian, there is an automated integration among both. An example of such integration is that once a specific hyperlink is placed from one tool to another, a two-way relationship is automated done (i.e. the link will appear in both applications although only one had the hyperlink copied into).

Project 3 uses Jira, Confluence and SharePoint, saving some documentation into SharePoint while others are place into Confluence. In project 3, the UX information (majorly mockups) were stored into SharePoint and a link to such was added into Jira's User Stories. Overall documentation was placed into Confluence, although some old documentation were on SharePoint.

These results of this early analysis help to answer the project RQ1 that stated: "How are UX information and USs connected into software virtual environments?". Given

the analysis, it could be concluded that there are usability elements used in the US life-cycle. And, given coding results, there are both implicit and explicit UX information used in the projects, where the *implicit* stands for information that could be presented in already known UX artifacts, but were only found in textual ways or within other artifacts that not a specific one for such UX information. The connections between UX information and USs though could be found in different ways (i.e. link, attachment, etc.) and locations (i.e within the US or store in other tools). Such aspects will be later explored in this project.

After these findings, given UX elements were found in the projects, another topic of investigation was done: *If the UX elements are used, how are they related to the US?*. This investigation relates to the master project RQ2: *"What are the navigational distances found to access UX information from USs into software virtual environments?"*.

Such question is even more important when considering works such as the Theory of Distances, presented by Bjarnason et al. (2016) and Bjarnason and Sharp (2017), which presents several types of distances found across a project life-cycle (refer to Section 2.6), as well as problems already highlighted by Liskin (2015), where manual links are often not created by project developers, given lack of time or, given many changes in requirements, the linkage management was too burdensome. Also, the work of Choma, Zaina and Beraldo (2016), which states that product owners (POs) have difficulties in understanding usability concepts, and also do not know how to incorporate UX issues in the product requirements. The same work also states that POs were most familiarized with USs to deal with agile requirements.

Due to such issues reported in the literature, and the fact that the coding analysis didn't answer all the questions of this study, the artifacts previously coded, had its dispersion analyzed. Such analysis is described in this section.

### 3.2.5.1 UX Elements Dispersion Analysis

Once the coding had been performed in both requirement documents and USs (refer to 3.2.2), given the results presented in section 3.2.4, a further analysis to understand how the UX elements were dispersed in the virtual element used in the projects was carried out. This analysis aimed to cover the RQ2: *"What are the navigational distances found to access UX information from USs into software virtual environments?"*.

The analysis had the USs as the initial point. Given that the USs were stored only in virtual tool (i.e. Jira), it was possible to track all the connections in the virtual environment having the USs as the starting point of such track.

We examined whether the US had connections to other virtual tools (e.g. Confluence pages and artifacts stored in SharePoint), as well as connections to other places within Jira



itself (e.g. connection to Epics, Tasks, or even other USs). Furthermore, we also analyzed connection with potential artifacts and UX elements stored in the US itself, through an attachment for example.

The connections here mentioned stand for any type of explicit relation through two or more artifacts or virtual environment pages. Such relation would need to cause a navigational distance. The connections among the virtual environment could be found through hyperlinks, attachment, mentions, and so on.

The artifacts positioning, in the virtual environment, was then structured in a graph-based structure, Figures 19 to 21. With such structure it was possible to apply the principles of the graph theory (HARRIS; HIRST; MOSSINGHOFF, 2008) in order to retrieve the distance among the information previously mapped.

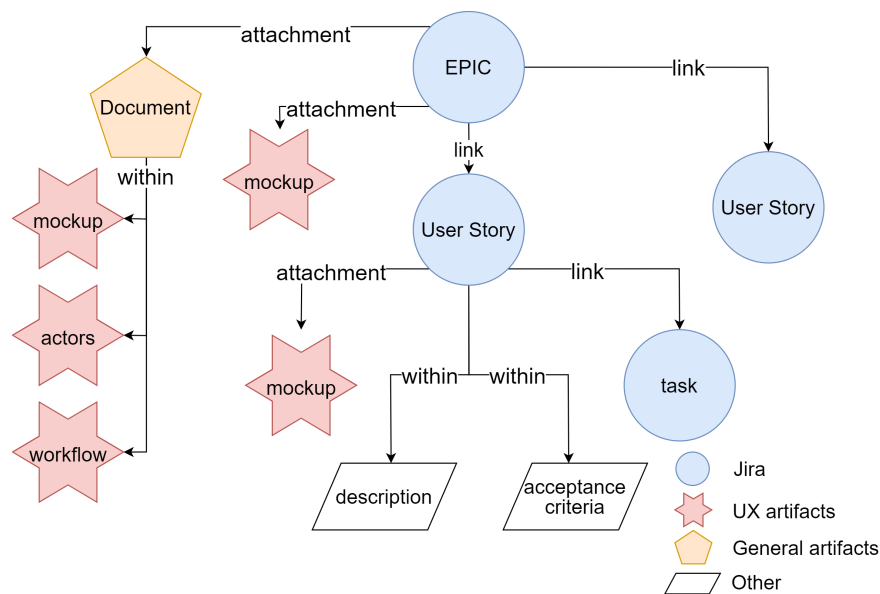


Figure 19: UX Elements dispersion example - project 1

Source: Author

The distance used in this study adds a classification to the type of navigation mentioned by (BJARNASON et al., 2016). According to (BJARNASON; SHARP, 2017), the navigational distance can be calculated from the number of clicks; this study proposes a classification for the type of navigation that is being performed, given that the navigation considers how the artifacts are related to each other (e.g. attachment or hyperlink). The type of navigation was based in the graph theory, from which the artifact/information distance is calculated by the distance between two vertices in a graph, being the number of edges in a minimum path connecting them (HARRIS; HIRST; MOSSINGHOFF, 2008). Moreover, if two artifacts can't be accessed through a path (i.e. they do not have a connection), according to graph theory, they are considered different components, making the distance between them to be infinite (HARRIS; HIRST; MOSSINGHOFF, 2008).

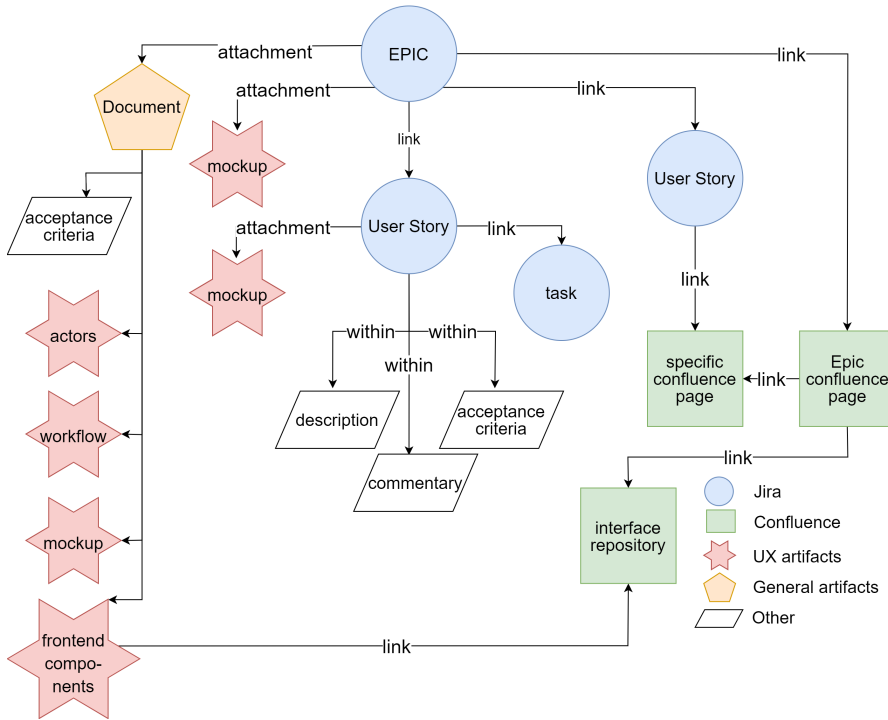


Figure 20: UX Elements dispersion example - project 2

Source: Author

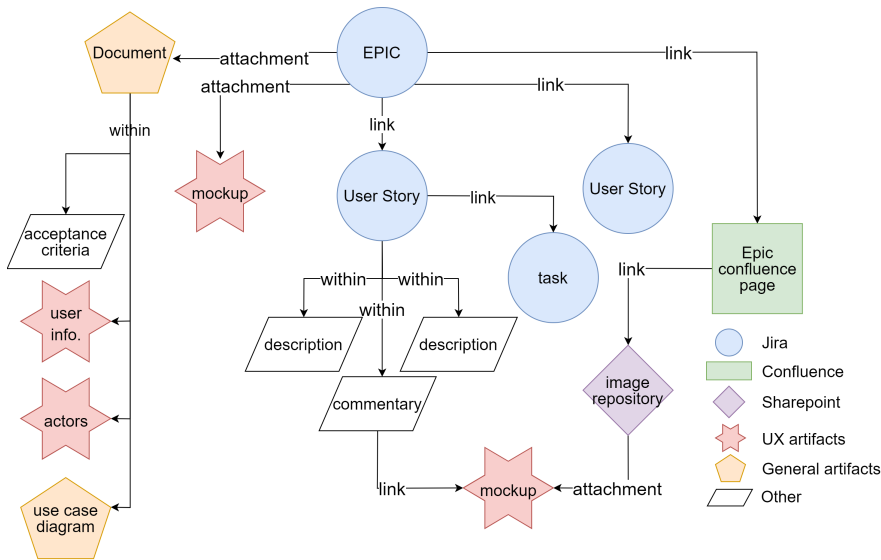


Figure 21: UX Elements dispersion example - project 3

Source: Author

### 3.2.5.2 UX Information Dispersion Findings

As a result of this information dispersion analysis, we mapped the connections among the UX information and other documents or artifacts into a graph-based diagram. From this mapping we were able to identify in which ways the UX information could be

connected to the US. Later, we found out different types of navigation distances.

Still considering the outcomes from the case study, we observed that the navigation happening in the virtual environments could be found in different ways. Bjarnason and Sharp (2017) describes the navigational distance as the number of clicks required to go from one information to another. From our observation, we concluded that these "clicks" could happen in different ways.

In Figures 19, 20 and 21, we can see where the UX information is in relation to other USs and other documents and artifacts, including UX artifacts, in the different projects we explored in this study. The connections between two elements (i.e artifacts, tools and documents) represent how information can be retrieved. We considered that such connections comprise the distance of one element to another. The navigation distance appears in the cases that individuals need to access multiple artifacts in a sequence to reach the information. The connections we found out are classified in three different types based on the fundamentals presented by (LISKIN, 2015).

**Within/Body** connection represents the minor distance possible to find information from a given artifact or document. This distance is present when an artifact or document is within another or when information is within an artifact or document but not as an external link or attachment. This type of distance can be considered a characteristic of a *container* artifact or document, which is an artifact that holds the information together in one place (LISKIN, 2015).

**Link** connection consists of a hyperlink to an URL or location in which the information is stored. Linking can be considered challenging when the parts to link are not isolated. The link creation/management may not always be properly done by developers. In cases that a developer is in the middle of an activity (e.g. coding), if they need to search for information, once they are found, s/he could prefer to continue in the previous activity instead of interrupting it to create artifact links (LISKIN, 2015). This could lead to the artifacts not having their relationship properly established, disrupting further information retrieval processes.

Finally, **Attachment** represents when an artifact is attached to another. This type of connection is commonly found in containers artifact (LISKIN, 2015). When working with container elements though, the attachments can be directly accessed. Such particularity makes the attached artifact to be an easy way to obtain detailed information. However, the attached elements can only exist within the container element and cannot be accessed otherwise (LISKIN, 2015). Therefore when comparing the same with links, this second option may be the best. However, the distance may increase due to the navigation to another environment.

The connection types here stated, and the mapping performed in the virtual

environments (Figures 19 to 21) were the basis for a navigational distances classification, based on the graph-theory (HARRIS; HIRST; MOSSINGHOFF, 2008). Such classification will be presented in Chapter 4. The types of information connection here exposed are a contribution to answering the RQ1 - "How are UX information and USs connected into software virtual environments?", outlined at the beginning of this project.

### 3.3 Case Study Conclusions

From the results from the case study, we present a summary of the conclusions, relating the same to the master project previously presented.

The presence of UX elements in all three analyzed projects could be verified, even though no project had a UX specialist in the team. The usage of UX information, initially present in the software requirement documents, could also be verified in the USs that were written based on such documents. Although the UX elements were found in US level, they could also be found spread through the virtual tools used by the teams. Given the task "hierarchy", the information could be placed at a higher level, which involves a higher scope. This can be exemplified by a UX information that is arranged in an Epic level. Such UX information would, therefore, be understood as being common information for more than one US under such Epic. These conclusions summarize how the UX information and USs were connected into the software virtual environments used by the teams of the case study, answer the RQ1 - "*How are UX information and USs connected into software virtual environments?*".

Given UX information arrangement into more than one virtual environments, and even in different places within the same environment, the presence of navigational distance could be verified. The UX elements dispersion diagrams (Figures 19 to 21) illustrate such navigation that needs to occur, given the UX information dispersion. During such analysis, some different types of navigation could be observed. Such types of navigational distances will be further described in Chapter 4, but such observations from the case study contributed to provide insights for answering RQ2 - "*What are the navigational distances found to access UX information from USs into software virtual environments?*".

## 4 Improving Navigational Distance

This chapter covers the findings and the recommendations from the analysis of the case study reported in Chapter 3. We present some potential issues found during the analysis, relating them to the ones reported in the literature (Figure 22 - Step D1). Next, focusing on the navigational distances found in the virtual environments, we present a classification system, which aims to help in the understanding of such navigational distances (Figure 22 - Step D2). This classification extends the work of Bjarnason et al. (2016), Bjarnason and Sharp (2017), which does not cover the types of navigational distances that may happen in virtual environments. Moreover, we also present a classification for the navigational effort, considering its execution in virtual environments.

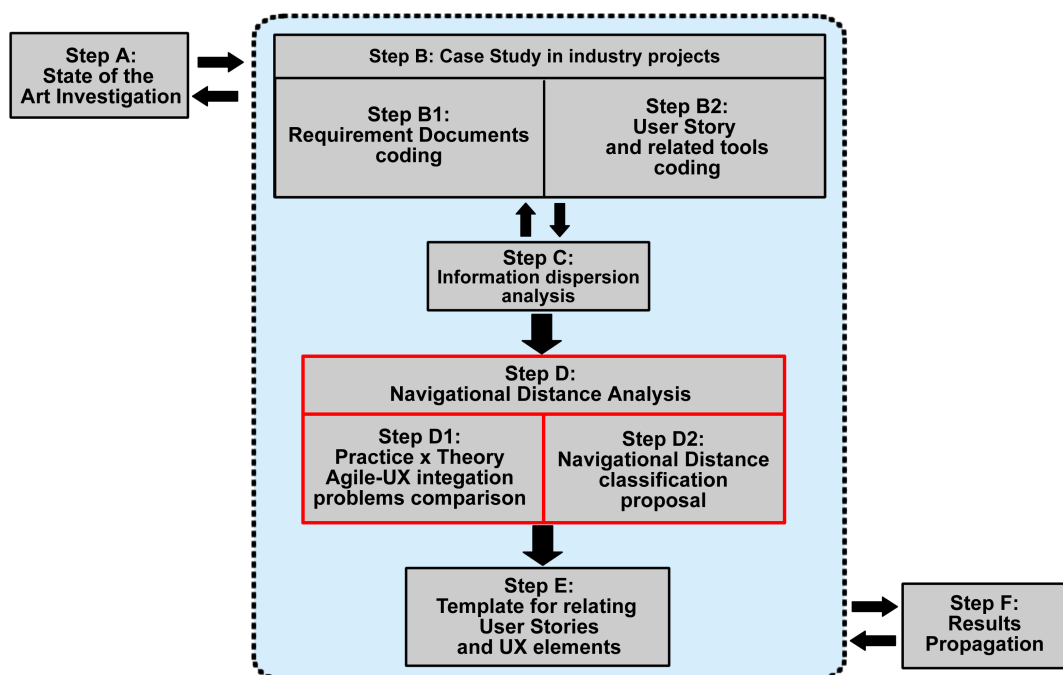


Figure 22: Navigational Distance Analysis

Source: Author

Considering the issues initially reported, as well as the classifications proposed, we present some good practices recommendations to avoid falling into the issues highlighted. We also present a template for the US creation and relation to UX elements, as well as its arrangement into virtual environments. Such templates aim to help in the decrease of navigational distances, considering the categories for the navigation and effort presented.

## 4.1 Practice and Theory Comparison

As a preliminary result of the analysis performed in this study, the following could be concluded: Although no UX specialist was part of the teams, UX elements could be found being used. Despite the US contained some UX elements within it, the arrangement of UX information outside the US was found. Such scenario allows the presence of different types of relationships among UX elements and US, being such already exposed under the Within, Link and Attachment types. The arrangement of the UX information in the virtual environments observed in the case study also indicates the possible presence of information traceability and navigational distance issues.

Taking such observations, and considering what has been already reported in the literature, here we present a comparison on the case study outcomes and the issues on UX-Agile integration currently stated in the literature. Table 3 summarizes the issues, also displaying some existing studies that already covered similar topics:

Id	Issue	Description	Related Works
I.1	Loss of big-picture	When information is spread in different artifacts and locations, checking only one does not provide the full view of the requirement	<a href="#">Silva et al. (2011)</a> , <a href="#">Garcia, Silva and Silveira (2017)</a> and <a href="#">Jurca, Hellmann and Maurer (2014)</a>
I.2	UX elements traceability	as the number of information and artifacts grow, the management and traceability becomes harder	<a href="#">Lee, Guadagno and Jia (2003)</a> , <a href="#">Kashfi, Nilsson and Feldt (2017)</a> and <a href="#">Silva et al. (2018)</a>
I.3	UX elements in agile practices	Neglecting non-functional requirements is considered as a major challenge for agile practices and can be the reason for lapse and rework	<a href="#">Rogers, Sharp and Preece (2019)</a> , <a href="#">Inayat et al. (2015)</a> and <a href="#">Jurca, Hellmann and Maurer (2014)</a>
I.4	UX information dispersion	When information is dispersed, the information duplication can become even more present	<a href="#">Deshpande et al. (2016)</a> , <a href="#">Sharp and Robinson (2006)</a> and <a href="#">Bjarnason et al. (2016)</a>
I.5	Navigational distance to UX elements	The increase of UX elements may increase the number of relationships with other artifacts. This will cause the navigational distance among UX and agile artifacts to increase	<a href="#">Bjarnason et al. (2016)</a> and <a href="#">Bjarnason and Sharp (2017)</a>

Table 3: Summarization of Issues found out

Next, each of the issues found out, reported in Table 3, are further explained. We present each of the issues by explaining it through a description, followed by a brief discussion of the same. Next, the impact of such issues regarding the navigational distance in virtual elements are reported, being such impacts described considering a developer's perspective.

### 4.1.1 Loss of Big Picture

It can be described as the loss of the holistic view of the project ([SILVA et al., 2011](#)). The sharing of documents, artifacts, and knowledge between the teams is a way

to keep this issue under control (SILVA et al., 2011). In the case study, the dispersion of the information, along with its arrangement into multiple locations within the virtual environment, could cause a problem in the loss of big-picture.

Artifact mediating communication is a topic already advocated in the literature (BRHEL et al., 2015; GARCIA; SILVA; SILVEIRA, 2019). Many requirements are used in software development practices, being the US the most used in agile practices (GARCIA; SILVA; SILVEIRA, 2017). USs are used for information transmission and task traceability, as well as software development progress (GARCIA; SILVA; SILVEIRA, 2017). USs are made to be kept small, so the pieces of work (tasks) can be done and delivered in a small period (COHN, 2004). Given requirements need to be broken into small pieces to fit the US structure, the information dispersion in many USs is a common scenario in agile practices. If a developer only take a look in the piece of work that needs to be done, without the acknowledgment of other tasks that may impact (or be impacted by) the task it is executing, problems may arise.

This is reinforced when UX elements are present in the development context. Given UX is considered a holistic field (HASSENZAHL; TRACTINSKY, 2006), the need to see the whole picture is important for a better understanding.

One of the impacts of loss of big picture is that developers need to have previous knowledge of where the task fit into the software development bigger picture (e.g. Epic). After mapping the UX elements and its location in the virtual environment, there could be seen a distance among the UX elements and the other task-related information, being this a contribution to the loss of big-picture (Table 3 I.1), as it is harder to see all information and its connections when such UX elements are dispersed.

#### 4.1.2 UX elements traceability

As mentioned by Lee, Guadagno and Jia (2003), in complex projects, management of the hyperlinks (or other ways of relationship) can become a difficult task, and its accuracy depends on the updating frequency. If relationships are not maintained correctly, hyperlinks from one artifact to another may be incorrect, as requirements or other requirements documents are disposed of. Moreover, Kashfi, Nilsson and Feldt (2017), Jurca, Hellmann and Maurer (2014), Silva et al. (2011), Silva et al. (2018), also reported that traceability between UX and business requirements is often lost in projects.

The coding performed in the 3 projects showed a variety of artifacts being used. The analysis of artifacts' relationship, which later led to the graph diagram, shows how such UX elements are arranged in the virtual environment. It can be seen that they are arranged in different tools and different levels within each tool.

This indicates the presence of *Requirement traceability* issues (Table 3 I.2). The

number of artifacts used in software development makes it difficult to manage, specially when the same are stored in different ways and places, without a proper pattern. Impacts of this traceability issues can be related to the US structure itself. Placing information in USs makes the requirement traceability difficult, given the number of USs used during all the project development life cycle. Moreover, the duplication of information is also common, and makes the information update harder, once that dispersed information will have to be updated in more than one place. Requirement traceability (and history recovery, in later steps of the development) can become difficult in large projects.

### 4.1.3 UX elements in agile practices

As stated in the literature, the usage of agile requirements engineering still faces challenges such as the lack of approaches to deal with non-functional information (INAYAT *et al.*, 2015).

Although the projects studied didn't have UX specialists, the usage of UX elements could be verified. The coding results showed that UX information were used in the projects, reinforcing Table 3 I.3 importance. Moreover, the UX information, initially found in requirement documents coding, were later found across virtual environments used by the teams. For example, a mockup attached in the requirement document was later attached to a US that related to that requirement, or a specific requirement regarding screen's component color described in the requirement document was part of the Jira card description or acceptance criteria.

From these findings we can highlight that, although there is no UX specialist in the projects, UX information was used, reinforcing its importance (JURCA; HELLMANN; MAURER, 2014), even though they are not the main focus of the requirements elicitation. Regarding the impact of such finding, the usage of UX elements without proper known of them may indicate that stakeholders involved in the project posses a tacit/implicit (ROGERS; SHARP; PREECE, 2019) knowledge of UX. On the developer perspective, development focused on UX aspects may be done without proper knowledge. The understanding of UX practices could help on better development and, consequently, better UX in the final product.

### 4.1.4 UX elements dispersion

The description of UX elements dispersion here states for its arrangement in the virtual environments. Where the dispersion is caused by the arrangement of UX elements into several places, may it cause information duplication or not.

Given agile structure, fast and small deliveries are made (BECK *et al.*, 2001; MATHARU *et al.*, 2015; DYBÅ; DINGSØYR, 2008). In order to be compliant with this,



small pieces of work are required, so they can fast analysed, developed, tested and delivered, so a feedback from users can also be taken fast (DYBÅ; DINGSØYR, 2008; INAYAT et al., 2015).

In order to keep tasks small (COHN, 2004), requirement decomposition is done (TAIBI et al., 2017), which will make information to be distributed into several artifacts. These will then be used by one or more tasks or US. Although a task is a self contained piece of development work, it may relate to other tasks, which other requirements.

In a top down view, this make the requirements to be dispersed into several US, being developed and delivered in different period of time. The analysis also showed that UX elements were not centralized in only one virtual environment, being distributed (Table 3 I.4) in several virtual tools.

Given the usage of virtual environments (DESHPANDE et al., 2016) and its importance on the information management and for the information storage and usage as central point for search, specially on distributed teams (SHARP; ROBINSON, 2006); this finding regarding information dispersion was used as the basis for the proposal on how to better relate the UX information with traditional agile artifacts, such as the US.

The impacts of the dispersed information is that, from such dispersion, the navigational distances mat occur. The presence of navigational distances is already mentioned in the literature as a possible problem in the software development/management processes (BJARNASON et al., 2016).

#### 4.1.5 Navigational distance to UX elements

Given the studies of Bjarnason et al. (2016), Bjarnason and Sharp (2017); it is understood that there are several distances between the ones involved in the projects and the information that needs to be communicated through different ways. Among such distances, the navigational distance could be analyzed in this study.

Its description is given by Bjarnason and Sharp (2017) as the number of clicks to navigate from a requirement to the test cases which verifies it. Here we expand such description, by the distances between any information. Moreover, the distance is not measured by the number of clicks performed, instead we propose it to be any navigational step occurred in the virtual environment, being it through a click or any other navigation mechanism.

In this study, the focus was in the navigational distance in virtual environments, given their usage in the projects. Moreover, the virtual artifacts and their supporting tools have already been mentioned by Deshpande et al. (2016) as being key information hubs for all team members in remote teams, a practice that is becoming more common given co-location is not always possible for various practical and business reasons (DESHPANDE

et al., 2016). This topic is more explored in the next section.

The impacts of the navigational distance to UX elements contributes to the mentioned issues of loss of big picture, UX elements traceability and UX information dispersion (Table 3 - I.1, I.2 and I.3)

## 4.2 Navigational distance classification

Considering the RQ2 - "What are the navigational distances found to access UX information from USs into software virtual environments?", and the insights the case study results provided on this subject, we now present a classification for the navigational distances, being such based in the case study outcomes.

Before presenting the navigational distance classification, and to better illustrate the distribution types later explained, we introduce Figure 23 to represent a fictional scenario that involves a Management tool and a Repository tool. The Management tool stands for a tool responsible for holding artifacts such as Epic, US and Tasks. The Repository Tool is where UX information will be placed, considering a wiki-based structure.

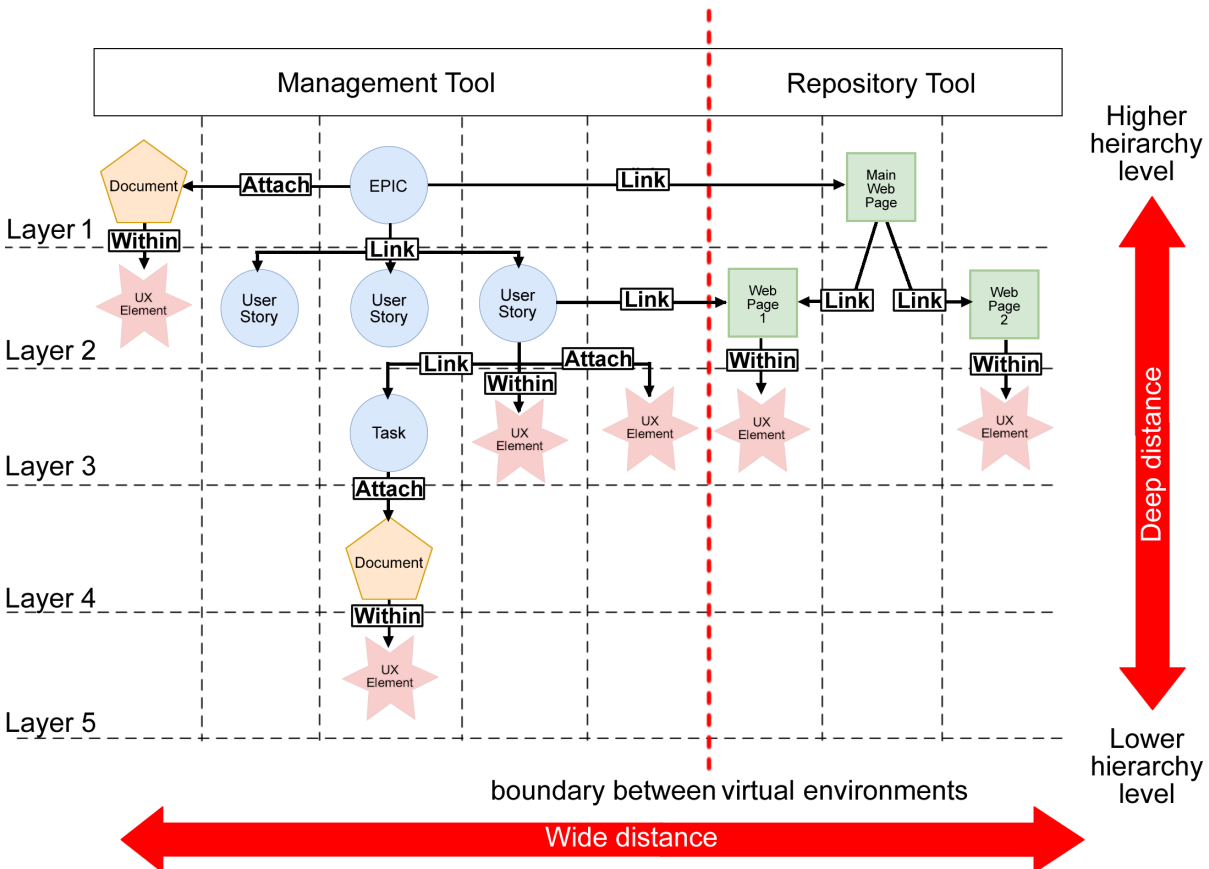


Figure 23: UX Elements dispersion sample diagram

Source: Author

As seen in Figure 23, having a unique US as sample, it is possible to verify the **Wide** distance, where horizontal navigation is made in the diagram. An example of such distance can be taken as navigation among two different virtual environments, such as from the US to a Repository page. This navigation could happen within the same tool. In such scenario, the navigation in the same "hierarchy level" would happen. For example, navigating from a US to another.

The **Deep** navigation can be seen when navigation is taken in a vertical format. An example of such navigation is when it occurs within the same tool, for instance, from an Epic to the US or from the US to a Task.

It is worth noticing that Figure 23 also covers an important aspect of the navigation distances here presented. Between Management and Repository tools, a red line can be seen. The same stands for the change of virtual environments. Some implications of having two or more tools related to each other need to be considered, as they have a direct impact on the wide distribution type. The link to another tool may require a new login to be made, or if the link does not directly point to a specific point in that application, a new search (even a simple one, as scrolling down the page) may be required, increasing the navigational distance from the initial point the user was to the final information being searched. Another issue on navigating through different applications is that, if one tool is to be changed by another, management of all previous relations, otherwise the same will not be valid anymore. In such scenario, even though an explicit link exists, if it is not valid anymore, the navigation it intends to provide can now be classified as the infinite type, once that developer is incapable of reaching information from such broken link.

Considering the concepts presented, illustrated in Figure 23, we now present the navigation distance classification, based on the case study results, which revealed that the distances (Table 3 I.5) among the artifacts used in the projects were not very large. This conclusion comes from the total number of navigational steps required to go to the farthest artifacts containing information that relate to each other. In the study, such steps amount was not big, varying from 3 to 6 navigational steps. This low number of navigational steps indicates that there is no need for many navigations through the virtual environments in order to find a given piece of information.

Considering the information arrangement observed in the case study (Figures 19 to 21), despite the distances among the information not present great depth, we believed that the graph-based distribution has the potential to become wide. This conclusion is based on the fact that one software requirement can generate many USs and for each US stored in a management tool, there could be connections to other virtual tools. In our study, the USs placed in a management tool had connections to repository tools or with other places within the same management tool. Moreover, USs may have interconnections among themselves. Considering that each US is within the same hierarchical level to each

other, connecting two USs would create a wide connection, considering the graph-based diagram early presented. Also, from the USs, connections to other virtual tools could be done, making the information distribution, initially concentrated in the requirement documentation, to be spread in a wide, but not deep, manner in the virtual tool.

From these results we could classify the information distribution in four types: Zero, Indirect, Deep and Wide. The main characteristics of each of the navigation types are summarized in Table 4. Next we provide a written description of each proposed Navigational distance type, as well as its pros and cons.

Type	Description	Characteristics
Zero	UX information is stored at a single artifact level (i.e. US)	No need for navigational distance as all UX information required is available in one place.
Infinite	No link among two or more artifacts	UX information is spread in different artifacts, but there is no explicit link among them.
Deep	UX information is stored within an artifact group but spread across different hierarchy levels	The more detailed information are stored in artifact stored at lower hierarchy levels (i.e storing information in Epic > US > Task > Attachment)
Wide	UX information is stored within an artifact group but spread across several places in same hierarchy level	Information is not stored in very detailed level (deep) but is spread in different artifacts of a same hierarchy level (i.e storing information across several USs)

Table 4: Types of Navigational Distances.

The **Zero** type can be taken as an extreme case, in which the zero means that there is no need to navigate through the artifacts, once that all required UX information is stored in the artifact level. For example, a text color that needs to be changed in a screen given a special condition. Such information could be stored in the US description or its acceptance criteria. Given no navigation is needed to be performed from the US to get such information, the same receives the zero classification type.

A pro aspect of the zero type can be described as, storing all information into once single artifact, making it self-contained. This means that a developer doesn't need to go anywhere to fully understand the information and where it fits into the project big-picture, decreasing the amount of time one spends on understanding a requirement. On the other hand, one of the cons of having this navigation type is that, if the information is to be used elsewhere (for instance, in another US), the same will have to be duplicated. Information duplication makes it hard to have consistency in case of updates, also impacting in the information management.

The **Infinite** distribution type can also be taken as an extreme case. The infinite type, named after a graph principle that represents the distance among two unlinked points (HARRIS; HIRST; MOSSINGHOFF, 2008). The infinite type can be described as a navigational distance that is infinite, given that there is no relationship (path, in a graph view) among the information. Considering this type, the navigation is dependent on previous knowledge from the developer.

In an infinite type, a pro aspect can be taken as the lack of need to maintain artifacts relationship as such are not available. Therefore no time is spent in artifact relationship management. The artifact management, in complex projects, can become burdensome, and accuracy depends on the frequency of updates (LEE; GUADAGNO; JIA, 2003). However, with no explicit relationship available, the system (virtual environment) lacks traceability, which is a con aspect of the infinite type. Along with traceability issues, the difficulty on history retrieval may also be present, given there are no explicit relationship among dispersed information.

The **Deep** distribution type can be described as the scenario where UX information is stored it a very detailed level, which means that the information present in an artifact is focused on few aspects, having specific details for a particular subject. Such type increases the number of navigational steps to reach the final information. A characteristic of this type is that the information arrangement, and the navigation, is performed within the same tool.

In such type, the granularity (i.e. as deeper as the UX information is stored, the more detailed it is) can be taken as a pro aspect. However, if upper information hierarchy levels have a good overview, the need to go down to the deeper levels is low. Storing UX information in depth may indicate the usage of only one tool, given that links to different tools would create a horizontal linkage instead of a vertical one. The usage of fewer tools tends to make information management easier. The con aspect of this type is the limited usage of tools, as well as the increase in the navigational distance (in a horizontal way), given that specific information is only found in lower levels of the artifacts hierarchy.

The **Wide** distribution type is described as the UX information being stored in different places within the same level (in a graph-based hierarchy), making distances to be spread in width instead of depth. A wide distribution type can indicate multiples tools usage or many UX information being stored within the same hierarchical level in the same tool. This arrangement may indicate a lack of information granularity.

Different from Deep type, the more detailed UX information is stored not on a deeper level inside the same tools, instead, they are placed in a separate location, specific to the type of information it relates to. This creates several places in which information can be stored, but each place is a collection of information of a given type. In other words, it can be said that the information is being grouped together, given a common aspect.

Grouping UX information together is a pro aspect of the distribution type, as it easier for the developer to remember where to find information, once that everything related to a common subject can be found in one place. On the other hand, the presence of information relationships makes maintenance to be required. Moreover, different tool usage (or different environments - pages, places within the same tool), creates extra work on management, especially if the tools do not have integration among themselves.

### 4.3 Navigational effort classification

Given the navigational distance classification, presented in Section 4.2, we now present a classification on the effort required to perform navigation in virtual environments. Table 5 adds a second classification to the ones previously presented in Table 4. While the navigational distance classification relates to the arrangement of information in virtual environments, the navigational effort classification relates to the actual effort expended by a developer while performing the navigation. Both (distance and effort) classifications were created to be complementary, where the effort classification aims to add weight to the navigation classification types previously exposed, by considering the effort variable attached to the navigation action.

Type	Description	Characteristics
Weak	No navigation through different platforms is required	All information is stored in one tools and therefore possible information relationship are all within the same tool.
Medium	Navigation through two different platforms	UX information is spread up to two different platforms, requiring some navigation to be happening through the platforms.
Strong	Navigation through three or more different platforms	UX information is spread in three or more different platforms, requiring some navigation to be happening through the platforms.
Broken	Navigation through two different platforms is broken or the person do not have access to it	UX information is spread in different platforms, but it is impossible to navigate through them, may it because the user does not have enough privileges or because the relation is no longer valid (i.e. outdated relationship)

Table 5: Types of Navigational Effort.

The **Weak** navigation effort classification: A navigation that does not require a

change of virtual environment. This happens when all the information is presented within the same platform (tool, workspace, etc.). For example, navigating through the software management tool, one can go from a US to its Epic in order to get more context or go to its Tasks, getting more details. There is even the possibility of navigating through USs, but in any of such cases, all navigation happens within the same platform and therefore no great effort in the navigational distance is applied on what comes to change of environment. The pros are that all information is within one single environment, if there is the need to migrate it, all information can be easily found and migrated. Furthermore, all involved in the project can have its access normalized within one single tool, and are aware that any information needed will, probably, be found in such environment. The cons are the dependency of the single virtual environment, which can have limitations depending on the team's needs. Also, the information organization is limited to the features available in that virtual environment.

The **Medium** navigation effort classification: A pattern that has navigation between two different virtual environments. As an example, navigating from a management tool (i.e. Jira) to a repository tool (i.e. Confluence) page makes a change in the virtual environment. The pros are that different tools provide different features that can best serve the needs of a team or a subset of it. Arranging information is not limited to a single virtual environment can provide, therefore it can be made in a more flexible way. The cons of this effort classification are among the ones previously stated, being a summary of such the need of a new login whenever navigation between the virtual environment happen; the search for information can become harder when compared to the **Weak** navigation effort, given the search may need to be done in two places if an information location is unknown; the relation between the environments (i.e. hyperlinks) needs to have management in order to keep them correct, avoiding broken or invalid links for example.

The **Strong** navigation effort classification: It is an extension of the **Medium** type. The same principles of the **Medium** effort are applied, but considering navigational distances occurring over three or more different platforms. Therefore the difference between the **Medium** and the **Strong** navigational effort types is the number of different virtual environments through which the navigation needs to occur. As more virtual environments, more navigation that requires crossing the boundary between tools (refer to Figure 23) is done. As a reminder, some of the cons of having links between two tools are: Need of a new login, broken or unavailable link in case the team moves from one of the tools, etc. Such aspects are present in the **Medium** effort classification, but are reinforced in the **Strong** classification, given the increase in the number of different tools being used. In other words, The navigational effort becomes stronger (harder) according to the number of different virtual environment being used.

The **Broken** navigation effort classification: Happens when a virtual environment

access is not possible to be made, being it for the lack of user privileges or other causes such as the application not being used anymore or even accessible at that specific time. No pros can be analyzed in this classification, while the cons can be summarized as the incapability of reaching information.

All the navigational effort classifications showed on Table 5 are an extension, or sub-classification, for the navigational distance types present in the Table 4. Therefore a Wide navigational distance could also be classified as Wide-Weak or Wide-Strong, depending on the information dispersion/relation in the virtual environment.

It is worth noticing that the **Infinite** navigational distance type necessarily means that the navigational distance effort is of the **Broken** type, given that both stand for a non-existing/impossible navigation occurring between two or more different platforms. On the other hand, not all **Broken** effort type comes from an **Infinite** distance. For instance, a valid relation that does exist among two different information place in two different platforms could make a navigation distance to be of the **Wide** type, but given a specific user does not have access to one of the two platforms, for this particular person, the effort is **Broken**, while for another user with access granted in both platforms the same **Wide** navigation would have a **Medium** effort. Figure 24 illustrates the navigational effort aforementioned.

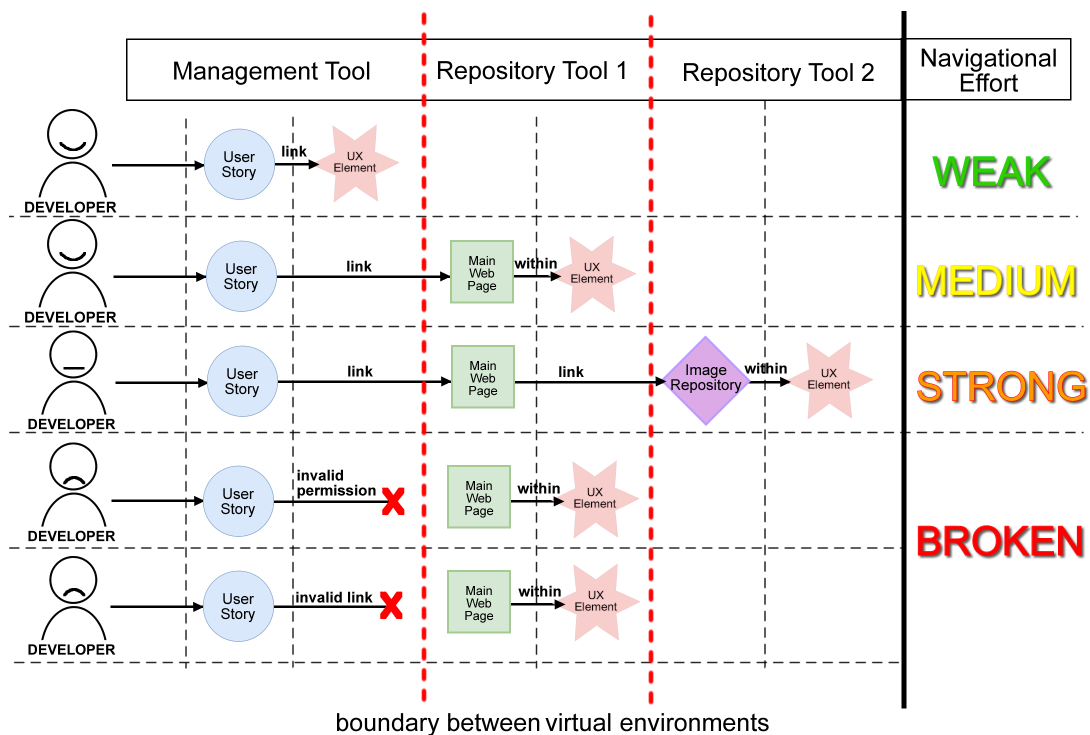


Figure 24: Navigational Effort

Source: Author



## 4.4 Reducing the navigational distance and effort

Given the findings resulted from the case study (Chapters 3 and 4), we now present recommendations summarized in Table 6. The recommendations have considered the issues mentioned in the literature review (Chapter 2) as well as the master project goals regarding the elaboration of recommendations to better relate UX information with USs and the proposal on how to place the UX information in virtual environments having the US as the central point of information search.

The recommendations are split into two categories: The first one, Subsection 4.4.1, proposes good practices that aim to help in the understanding of a few concepts that have been exposed throughout this master work; The recommendations also aim to help on decreasing the occurrence of some issues found during the case study and the literature review, summarized in Subsection 4.1. The second one presents templates for the arrangement of UX information in the virtual environment, covering the US creation and how to relate it to UX information.

### 4.4.1 Good practices recommendation

Table 6 summarizes each of the good practices recommendations, relating each to one or more issues that we have exposed in Table 3 (Section 4.1) and also present some of the related work used during the recommendation elaboration.

Next, an explanation to each recommendation is presented, having a description, a contextualization with the literature, and consideration of its impacts regarding navigational distances.

Id	Recommendation	Related Issue Id	Related Works	Affected Navigational Distances
R.1	Store generic information in higher artifact hierarchy levels	I.1, I.2 and I.3	Liskin (2015) and Lee, Guadagno and Jia (2003)	Deep
R.2	Link information across different virtual environments	I.1 and I.2	Lee, Guadagno and Jia (2003), Lucassen et al. (2015) and Deshpande et al. (2016)	Infinite, Wide, and Zero
R.3	Central repositories and Specific repositories creation according to artifact type	I.2, I.3 and I.4	Liskin (2015), Deshpande et al. (2016), Schön et al. (2016) and Shukla, Auriol and Baron (2011)	Deep and Wide
R.4	Use tools that best serve the specialists (i.e. don't stick to one tool)	I.2	Soares et al. (2015), Liskin (2015) and Silva et al. (2018)	Wide
R.5	Navigational distance awareness	I.5	Bjarnason et al. (2016) and Bjarnason and Sharp (2017)	Infinite, Deep, Wide and Zero

Table 6: Good practices recommendation.

**Keeping the generic information in higher levels of artifact hierarchy to avoid information duplication and avoid big-picture loss (R.1):** This practice

aims to create the understanding that, given an artifact hierarchy (Epic > User Story > Task, for example), storing information in the higher levels of hierarchy should be considered, when the same can be used to more than one artifacts that are in hierarchical levels below it.

Given the artifacts categorization described by [Liskin \(2015\)](#), container artifacts should be used on the higher hierarchy level. Arranging information in medium-low hierarchy level artifacts (e.g. US and Task respectively) is recommended if the information is specific to such artifact. Otherwise it can cause the need for information duplication, making information management a harder process ([LEE; GUADAGNO; JIA, 2003](#)). Generic information should, therefore, be stored in the top of the artifact hierarchy.

Such information arrangement has a direct impact in the *Deep* navigational distance, given the need to go down into lower hierarchy level artifacts is decreased in case the upper hierarchy levels already contain enough information for the software requirement understanding.

#### **Linking the different information stored across the virtual environment**

**(R.2):** Given the importance of virtual environments ([DESHPANDE et al., 2016](#)), this recommendation aims to create a common-sense, in which the information flowing in the virtual environments should be linked together, to facilitate its finding, as well as the navigation itself.

This recommendation also aims to reduce the loss of big-picture ([JURCA; HELLMANN; MAURER, 2014](#)). Although it may be difficult to have an understanding of the software requirement scope big-picture looking into dispersed information, having them linked to each other will make the navigation, and information finding, an easier process.

Given USs are one of the most used artifacts by developers ([LUCASSEN et al., 2015](#)), having one of its principles as to be small enough so it can be developed in a small period ([COHN, 2004](#)), the information being dispersed into several USs that derive from the same requirement may be an expected scenario. Overloading US with too much information may cause it to hold information that could be used elsewhere. Therefore placing information outside US is fine, but if the same related to it, a link should be made.

Linking all information will make the information to be found more easily, also helping in the information management ([LEE; GUADAGNO; JIA, 2003](#)). This recommendation can also be related to the US quality framework proposed by ([LUCASSEN et al., 2015](#)). In this framework, one of the quality aspects is called *pragmatic*, which, among other attributes, evaluates the linkage of all unavoidable, non-obvious dependencies for that US, making it more complete in an understanding point of view ([LUCASSEN et al., 2015](#)).

This practice has an impact on *Wide* navigational distance, once that information

stored in different tools, as proposed in this master project (Subsection 4.2, cause an increase in the Wide navigational distance. The practice of linking the information though, also has an impact on making the navigational distance to not fall into the edge cases of *Zero* or *Infinite* navigational distance types, once that the creation of the information link will make such scenarios to not happen.

**Create central repositories (R.3):** This recommendation will be further expanded a template recommendation (Subsection 4.4.2). The central repositories serve as central points where a given type of information (i.e. UX information) can be arranged. The proposal is of a wiki-based virtual environment, creating a common place for UX information flow and sharing.

The central repositories recommendation is based on the *repository* artifact type proposed by Liskin (2015), which is the type of artifact that holds other information (i.e. a repository). The creation of such repositories makes it easier to find information if you know to what type it belongs. The information will not be stored in several US (COHN, 2004), instead it will be centralized in the repository, and linked to the other artifacts (i.e. USs). This makes the navigational distance (BJARNASON; SHARP, 2017) to these information to have a limited maximum distance (once that there is a final point to be reached). The information management, history retrieval, etc. is also easier. (SHUKLA; AURIOL; BARON, 2011). Central repositories can also be seen as "landmarks" in the virtual environment navigation, making it easier to remember and find a initial step for a more detailed search for an specific information (VINSON, 1999). Moreover, it is possible to make a parallel of such repositories with the "UI toolbox" mentioned in the work of Schön et al. (2016), in which it is stated that these toolboxes included UI elements concerning their UX. The usage of such toolboxes relieved the UX experts and increased consistency among interaction design (SCHÖN et al., 2016).

Central repositories may also help on what is called information hub, described by (DESHPANDE et al., 2016) as central places where information flows meet and decisions are made. With central repositories, it is easier to find information and the same are organized according to some category (i.e. UX-related, database-release, etc.). The repositories can also serve as buffers, being described by Deshpande et al. (2016) as where information is held until it can be processed without disrupting ongoing activity. By storing UX information outside the US, while keeping their relationship explicit, fulfills a US quality aspect from the quality framework proposed by Lucassen et al. (2015). The *syntactic* quality aspect aims to keep USs atomic, minimal and well-formed, not directly overloading it with too much information. In order to keep the possible development dependency on the UX information, a task could be created within the US, making it clear that the same is only to be fully developed once such UX task is completed and properly linked to the US.

The repositories have an impact on both *Deep* and *Wide* navigational distances, given that some information previously stored into USs (or high/lower hierarchy levels compared to it) can now be placed into the repositories. This helps in the decrease of *Deep* navigational distance, but increases the *Wide* navigational distance. Moreover, the creation of a repository in a different, wiki-based, tool, adds the *Medium* effort to the *Wide* navigation. Despite that, we believe that the aforementioned pros of using the central repositories are enough to supersede the cons of adding the *Wide-Medium* navigational distance-effort along with its downsides (refer to Tables 4 and 5).

**Using Tools/artifacts according to the team needs (R.4):** Is a recommendation that, given the exclusive usage of a given tool, the same may not best serve all the needs of each team member in an agile approach, which tends to be cross-functional. This recommendation aims to incentive the usage of more tools, which best serve the needs of each specialist in the team. It is important to have in mind the other recommendations here exposed. Therefore the usage of multiple tools can be done, as long as the navigational distances do not increase. The usage of repositories for information centralization can have an important impact into this recommendation.

If USs are not enough for the teams involved (e.g. developers, UX specialists, QA specialists, etc.), to use other tools (specific for each specialist) and just link US to information is a way to diminish problems already reported in the literature. Experiments with questionnaires identified that User Stories may not be sufficient to support all development activities, such as time estimation and software design (SOARES et al., 2015). On the other side, interviews indicate that often a variety of artifact types is needed to successfully conduct a project. At the same time, using multiple artifacts causes problems like manual translation effort and inconsistencies (LISKIN, 2015). Moreover, often, there is not one perfect kind of artifact that will serve the needs of all participants so that the project needs a whole variety of different artifacts (LISKIN, 2015).

When trying to merge agile and UX, to allow each specialist to use its tools seems to be a good approach. The merge should not be at a tool level but in a link level. Creating links among artifacts helps by: 1) Improving US details (which when low may difficult the task development) while also keeps user story small, as information will into be stored withing it (e.g complementary info is stored outside the US) (COHN, 2004). 2) Diminishing the difficulty in handling information management and traceability (LEE; GUADAGNO; JIA, 2003). This recommendation is also based on what has already been concluded in the work of Silva et al. (2018), where the problem of combining UXD and agile methods can be taken as an example of a context-dependency issue. Different teams in different contexts use different artifacts and techniques to create a shared understanding. Therefore tools are needed to support developers in acquiring and sharing UXD and software engineering best practices. The tools should also be flexible enough for developers to fit them into

their particular project context. This fits in what was stated by [Silva et al. \(2018\)](#), who concludes that Agile UXD will be more seriously considered if a computer-assisted usability engineering platform is available.

The impact of this recommendation is on the *Wide* navigational distance, given the usage of multiple tools may increase the width of the navigational distance, instead of its depth. This recommendation though should be done considering the other recommendations, so that the *Wide* distance is not increased beyond a manageable level. The usage of this recommendation along with central repositories (R.2) should be considered in order to keep the navigational distance width within acceptable limits.

**Navigational distance awareness (R.5):** To have the navigational distance in mind when working in virtual environments is the last recommendation. This is a subjective topic but, as a recommendation, it aims to create a common-sense to the ones using such environments, to understand what a miss of linkage among information arranged in different locations in different virtual environments may cause. The navigational distances categories and navigational distances effort proposed in Sections 4.2 and 4.3 will depend on the management done by each individual using the virtual environments, therefore the awareness of navigational distances is to have in mind the results your actions (or missing of actions, in the case of a link not being created) can have.

Given our analysis in the case study and the US structure, we concluded that, for the scenarios studied, the navigational distances are not a problem, provided they are kept to a manageable level. The teams need to find a half-way on task size (not too big or small) ([COHN, 2004](#)), keeping the "generic information" in higher levels of artifact hierarchy to avoid information duplication and avoid big-picture loss. Avoiding duplication reduces technical debt, facilitates information management and retrieval ([SOARES et al., 2015](#)). Storing information at proper levels helps in avoiding loss of big picture ([SCHÖN et al., 2017](#)) (e.g a information that is shared among many US under the same EPIC, doesn't need to be stored in US itself. The other way around is also true).

This recommendation has an impact on all navigational distances categories (*Zero*, *Wide*, *Deep* and *Infinite*), given that if not properly done, a relation, or miss of relation, among information in virtual environment can cause Infinite distance to happen. Placing too much information within one single place (i.e. US) causes the Zero distance to happen, but may impact other things such as the need for information duplication or the difficulty in breaking a US into small tasks ([COHN, 2004](#); [LUCASSEN et al., 2015](#)). Other information arrangements cause the *Wide* and *Deep* distances, along with the effort categorizations they may have.

#### 4.4.2 Template proposal for information arrangement in virtual environments

Considering the usage and arrangement of UX information in virtual environments, having the US as the main used by agile developers (SILVA et al., 2011), we now present some templates for US creation. The template will also cover how to include UX aspects into the US creation, by suggesting a process for adding UX information into US, in which UX-related tasks are created under an US, and enrich the same along with the task conclusion. The template proposed aims to complement the good practices presented in subsection 4.4.1.

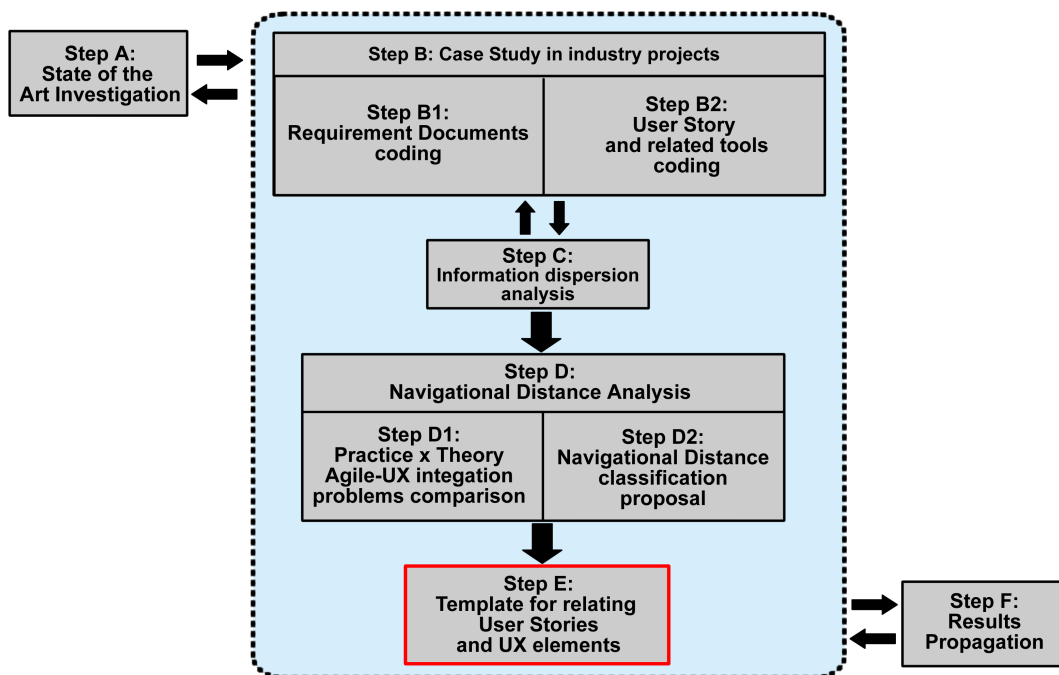


Figure 25: Template proposal

Source: Author

Regarding the UX information arrangement in virtual environments, we also propose the usage of a different environment from the one the US is stored. This second environment will serve as a repository, structured in a wiki-based format. The repository will be UX-centered, being the UX specialists responsible for such environment management.

The template for UX information arrangement in the virtual environment was designed based on the principle that UX information is present within agile projects, but is found dispersed across several places and in different ways in virtual environments, as per conclusions drawn in the case study presented in this work. We also considered the problems presented in the literature, such as by Kashfi, Nilsson and Feldt (2017), which concludes that there may be communication and collaboration gap between UX and non-UX practitioners, and Brhel et al. (2015), Garcia, Silva and Silveira (2017), Garcia,

Silva and Silveira (2019), Silva et al. (2018), that exposes challenges on integrating UX in agile practices, already reported in section 2.7.

Moreover, the template creation considered the navigational distances proposed by Bjarnason et al. (2016), Bjarnason and Sharp (2017), as well as the proposed categorization of navigational distance types and effort presented in Sections 4.2 and 4.3. Next, we present the templates and recommendations in the following order:

US template: We present a template for US writing, already considering how to relate it with UX elements, being it directly added in the US when the same is not to be used in other USs, or adding in a repository, adding links to such environment in the US.

UX Elements arrangement in virtual environments: A step-by-step guide regarding the creation and enrichment of the US with UX elements, as well as the arrangement of the same in the virtual environments. The virtual environments will be classified as Management tool and Repository tool. The Management tool stands for a tool responsible for holding artifacts such as Epic, US and Tasks. The Repository Tool is where UX information will be placed, considering a wiki-based structure.

Repository template: Given the arrangement described in the UX elements arrangement in virtual environment, the relation among Management tool and Repository tool are made. Therefore we also present templates on how to better organize the repository, considering the UX information that it will hold.

#### 4.4.2.1 US template

Beginning by the US, in order to keep it small (COHN, 2004), but with the aspects that make it a good US (LUCASSEN et al., 2015), we recommend the template shown in Figure 26. This template, it can be seen that both Acceptance Criteria and Description should be the focus of the textual UX information. Such textual information is kept in our recommendation given it was found during the coding process in all three projects. The US should contain a link to its higher-level hierarchy (i.e. Epic), and we also propose the creation of Tasks for UX-related work that needs to be done along with the US, having such recommendation being based in good practices by Lucassen et al. (2015).

Figure 26 is the template of a US with UX elements within it, as well as external links to UX information that can be stored in other tools. Figure 26-A is the links to the US Epic. This aims to give a proper placement of the current US within a bigger picture of the project. Also, in case there are not enough contextualization into the US, going to its Epic could provide such information.

Figure 26-B is regarding acceptance criteria. This is an important aspect already covered in the work of Choma, Zaina and Beraldo (2016), which included UX aspects into US and its acceptance criteria. During the case study, it was verified that in some cases

## User Story Description

### Details

Type: **Story**

Status: CLOSED

Priority: Medium

Fix Version: 2020.1

Label: None

Epic Link: [My EPIC](#)

Story Points: 5

Acceptance Criteria:

Acceptance criteria with aspects that should be covered in order to validate the User Story. Includes functional and non-functional aspects, including UX information.

These criteria may be used by Quality Assurance team in order to validate the User Story during testing phase.

Example:

- All fields in the form should be mandatory.
- When a field is not filled, a red warning message under the component should be displayed.
- When clicking in save, a confirmation message should be displayed.

Consists of:

**TICKET-111** - Task 1

**TICKET-112** - Task 2 (create UI prototype)

Mentioned in: [Central Repo. Page](#)

### Description

Overall description of the User Story. May include UX information, if not already in Acceptance Criteria.

UI Aspects:  
Screen prototype: [Link](#)

Figure 26: User Story static template for information organization

Source: Author

the UX information was placed as part of the acceptance criteria. We believe that, as long as the information is necessary for the conclusion of the US, adding it in the acceptance criteria not only make the navigational distance to be reduced to *zero* as well as gives something to be later evaluated in order to accept the final result of the US.

Figure 26-C represents the break down of the US into smaller tasks, a topic already covered in the literature (COHN, 2004; LUCASSEN et al., 2015). Here we propose the creation of an specific UX task, which is to be done along with the US. Such task will be responsible for all the arrangement of UX information into the virtual environment (see Figure 29) as well as to create a relation between the UX information created and the US from which it was originated (Figure 30). Figure 26-D and Figure 26-E represents the relation between the UX information and the UX. The first one is a link to a high-level page into an UX central repository, while the second is a direct link to a specific UX artifact. Note that by creating such direct relations, the navigational distance tends to be small, i.e. once the link is opened, the developer should access the final information,



without the need for further navigational steps.

Provided that the templates here proposed will consider the US as the main central point for information retrieval, the adding of information into US as presented in Figure 26 requires a tool that provides a set of features more complex than the ones provided by basic/free tools (for example Trello <sup>1</sup>). Therefore we recommend the usage of the US template in tools that contain features that allow the creation of a similar structure presented in Figure 26.

When working on basic tools though, the main point of the template is still applicable. Adding links from a virtual location to another is possible in any tool that accepts text input. Although in basic tools this task may be harder to be performed and to have track of. Considering the minimum aspects from the US template that we would recommend to add in tools, the relationship between the US and the UX elements can still be done (Figure 26-E). Provided the limitation that some tools may have, apart from the linkage on US-UX elements, done through a hyperlink, we also recommend the usage of the Acceptance Criteria with UX elements in it (Figure 26-B), as this will be used to validate whether a US is completed or not. The usage of the proposed template is still to be better evaluated, and its usage in different tools is a topic to be considered. The template proposed will consider tools that allow the creation of a US similar to the one presented in Figure 26.

#### 4.4.2.2 UX Elements arrangement in virtual environment

The recommended process of enriching the US with the UX elements, as well as the arrangement of such elements in the virtual environment, is presented in Figures 27 to 30. The Figures represent a step-by-step on how to use UX elements along with US writing process, by creating a specific UX task below a given US. The creation of repositories to better distinguish the information is presented.

We recommend the usage of two distinct, but related, virtual environments. The first one is the Management Tool, in which the artifacts such as Epic, US and Tasks are placed. The second one is the Repository Tool, which serves as a container (LISKIN, 2015) for holding different types of information in a wiki-based structure. The relation between the two virtual environments may create an extra effort on the navigation, already covered in Section 4.3. Such effort is represented in Figures 27 to 30 by the red line vertically crossing the two environments, indicating the boundary between both virtual tools being used. As already presented in Section 4.3, the cons of having two virtual environments cause the effort in the navigation to be considered Medium. Given the proposal next presented, we believe that the gains of having two virtual tools for a better information organization overwhelm the cons of having a Medium effort classification. Notice that,

---

<sup>1</sup> Trello: <https://trello.com/>

given description presented in Section 4.3, the limit for a Medium effort classification is the usage of two different virtual environments, which the proposal will be based at.

Starting by Figure 27, given a requirement that is big enough to be set as an Epic, we recommend the Epic creation along with a proper Repository specific page. This central repository page will be Epic-related.

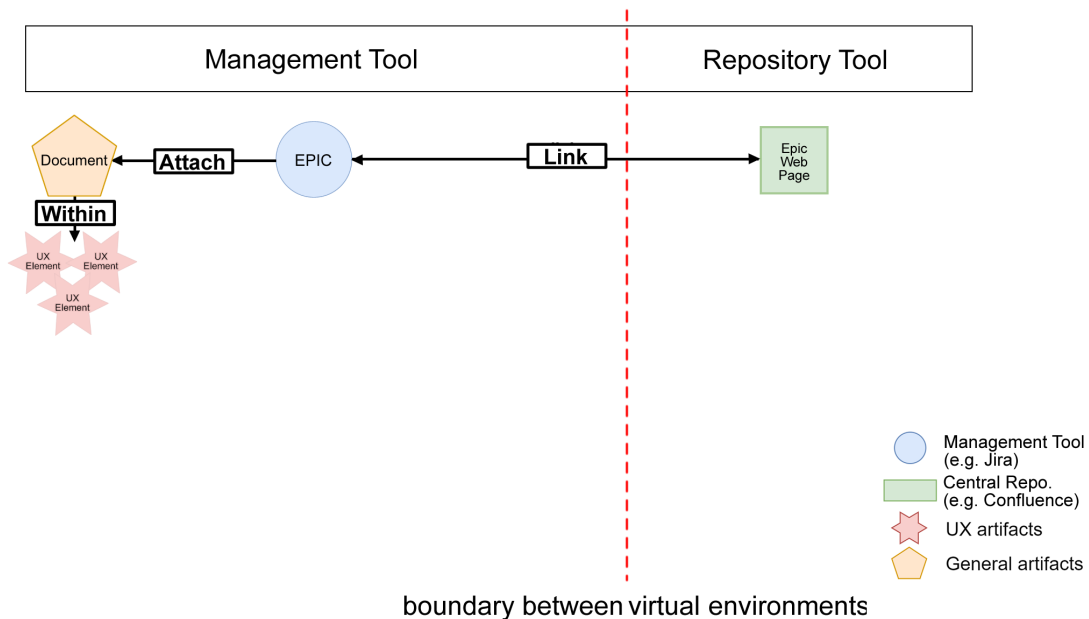


Figure 27: Virtual Environment Step 1 - Epic and Main Repository page creation

Source: Author

Once the initial Epic and specific repository pages have been created, the USs are created. Figure 28 shows such process for one US. Along with the US, the same is broken down into smaller tasks (COHN, 2004; LUCASSEN et al., 2015; TAIBI et al., 2017). Considering the existence of UX work to be done along with the US development, we propose the creation of a proper "UX task", which should be created under the US it relates to. Given tasks should be done prior to the actual US development (COHN, 2004), it is safe to state that a UX task should be done before the development of the US, but not too far from it. This process brings the UX closer to the actual software development, considering it to be done with the US.

Although showing the process for only one US, and given USs may be created under one Epic, the template proposed applies to each US. This way, each US would have a UX task underneath it.

Figure 29 shows the creation of a sub-section in the Epic (specific) repository page, being this section related to the UX work. The UX artifacts, and all UX-related information that is generated through the UX task development, should be arranged in

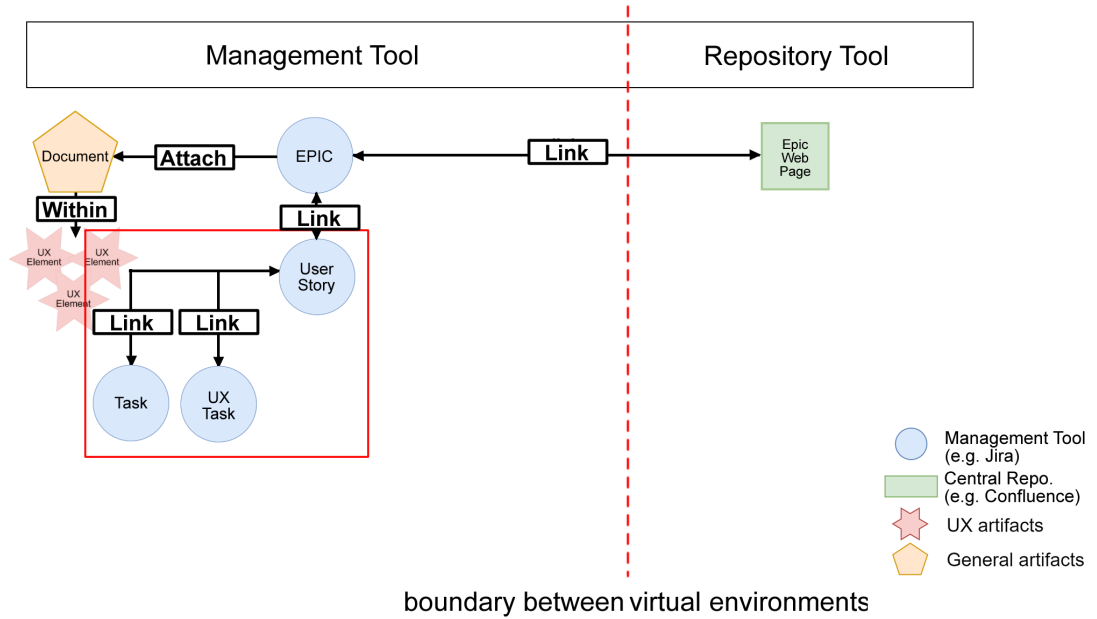


Figure 28: Virtual Environment Step 2 - User Story and UX Task creation

Source: Author

the repository page, under a section that provides a relation to the US it originated from.

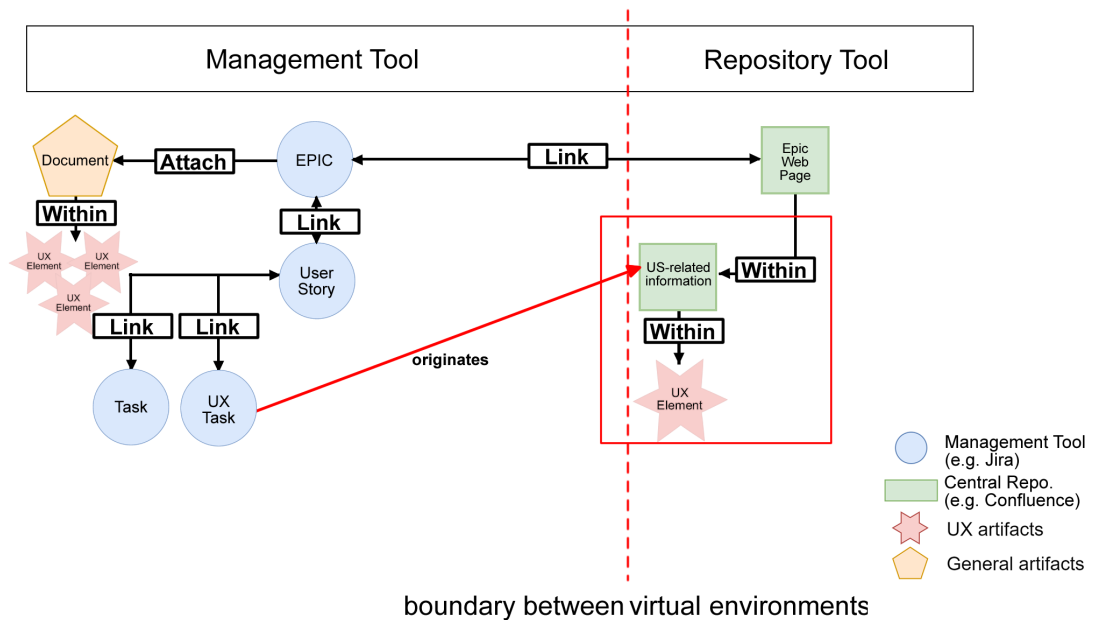


Figure 29: Virtual Environment Step 3 - Specific Repository page creation

Source: Author

Once the UX work is done, the US is enriched with more information (i.e. acceptance criteria or more detailed description). A link to the created repository page is also created,

making a relationship between the US and the UX information stored in the UX repository. This process is illustrated in Figure 30.

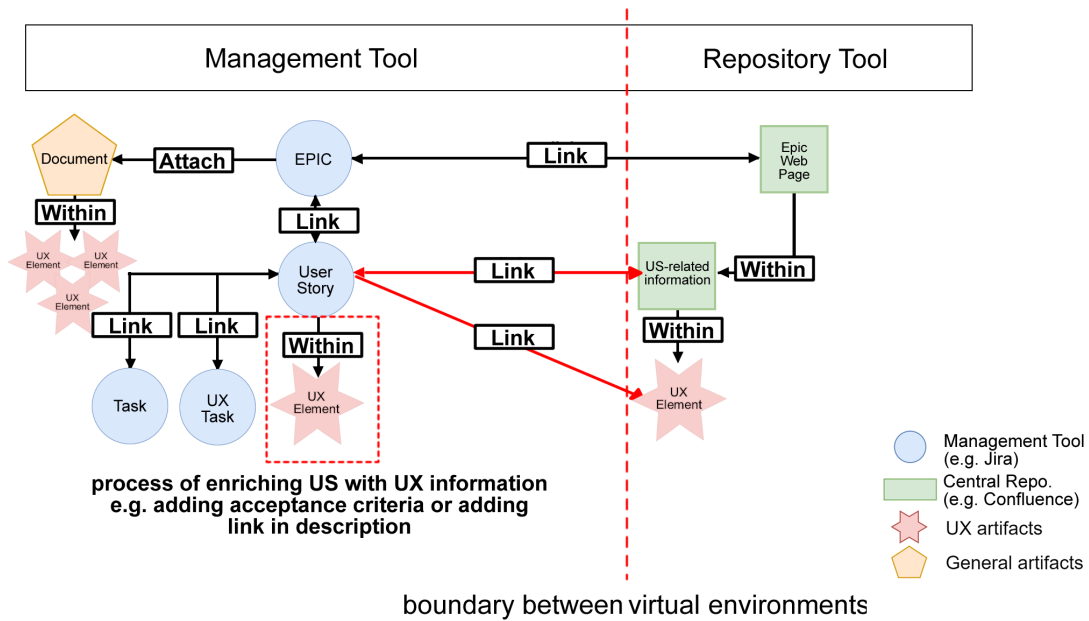


Figure 30: Virtual Environment Step 4 - User Story enrichment and link to UX repository

Source: Author

Although having a repository for storing UX information, the need to create a place in which information can be used in more than one US may arise. Considering such scenario, next we provide an overview of the Repositories, classifying it into two types: Central and Specific.

While the Specific repositories are for storing UX information that are related to a specific Epic (therefore the name specific repositories), the Central repositories aim to hold information that will be used throughout the whole project, being applied to more than one Epic.

Through a UX task, the need to add such "generic" information in the repositories may happen, and in such scenarios, the usage of the Central repositories is recommended. In such cases, if the usage of such information is also important to a given US, the linkage between a US and the Central repository may be done, the same way it would have been done in a US-Specific repository relation. This process illustrate in Figure 31.

We believe that the creation of US-UX repositories relationship makes the navigational distance between the US and the UX element to be, the maximum of, one. This means that the number of navigational steps to be performed from the US to reach a UX information is Zero, if the UX information is held in the US itself, or it is One, meaning that it is in one of the repositories. Being it a Specific repository or the UX Central

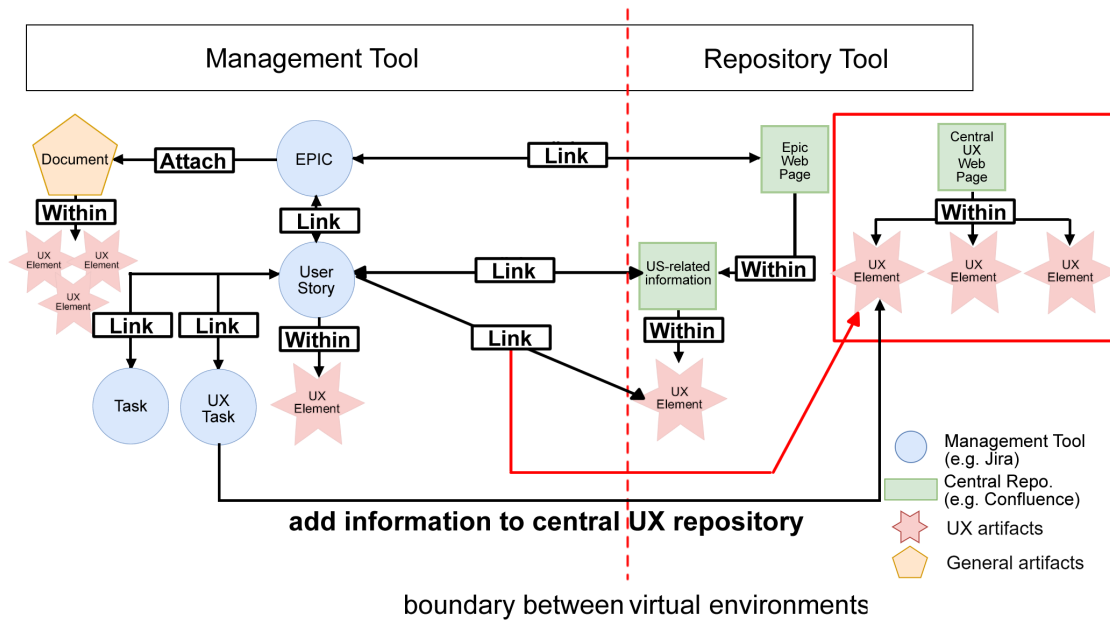


Figure 31: Virtual Environment - User Story link to UX central repository

Source: Author

repository doesn't make a difference from the navigational distance point of view, provided that direct links between them and the US are created.

Apart from creating a limit to the navigational distance, the process here recommended also makes the UX specialists to work closer to the developers, having them work under the same US. This may help on decreasing the distance between UX specialist and developers, reported by [Silva et al. \(2011\)](#), [Silva et al. \(2018\)](#).

Next we present a deeper view on the repositories here presented, also proposing templates for its organization.

#### 4.4.2.3 Repository template

Given the arrangement and dispersion of UX information in the virtual environment observed during the case study, the creation of a UX central repository is recommended. The same were based on the container artifact type described by [Liskin \(2015\)](#) as artifacts that contain other artifacts or its information. The repositories aim to create a common place of UX information storage, creating a common sense to the ones involved, to where to look up in order to retrieve a given information. Also, by creating a relationship to the repository, helps on reducing the navigational distance. This is because the repository can be considered, as here proposed, a final place for UX information storage. Therefore, no further navigation needs to be done once the UX repository is reached, once that all required information should be available in there.

Here we present the templates for the creation of two repositories: The Specific repository and the Central repository. The Specific repository is to be used within a limited scope. For instance, information that belongs to a single Epic should be stored in a Specific repository, which would be created specifically for such Epic. Therefore, the presence of several Specific repositories will happen through time. Given the information held in each Specific repository is, as the name suggests, specific to an Epic (i.e. specific to a given development), no UX information duplication among Specific repositories should happen. For UX information that is to be used among several Epics, the usage of a Central repository is proposed. In such repository, it is stored UX information that is to be used across the whole application, not being specific to a single page or component.

Figure 32 is a sample of a Specific Repository page (early presented in Figures 27 to 30). Figure 32-A is an overview of the sub pages that constitute the specific repository. Among many, a UX specific page is recommended, once that specific UX information related to an Epic can be stored in a proper place.

Given this specific repository page relates to an Epic, attaching the software requirement document in the specific repository (Figure 32-B) is recommended, once it enriches the repository with the essential information it relates to. Figure 32-C represents the links to the Epic and all USs that relate to this repository. By doing so, the virtual environments are related, and the *infinite* navigational distance will not occur (considering that the link will be also made from the Epic and US to the specific repository).

Figure 32-D displays a UX page under the wiki-based repository. The UX page refers to Figure 34 later presented. This menu will direct the user to the UX central repository. Also note worth, Figure 32-E shows that for many "Epic pages" are created within the repository. As presented in previous templates, the relation from an Epic to a proper Repository page should be done, considering that the USs done within the same Epic are related to each other. Please notice that the other pages under the menu are merely illustrative. Also, the creation of the menu structure may not be possible depending on the tool being used. The main focus of the template of the repositories is to have a place to centralize information, therefore the menu presented in the template can be changed.

Under the Epic page, specific pages should be created according to the needs of the project. Figure 33 shows a page with 3 sub-pages, being a Database, an Infrastructure and a UX page. For this study, given UX is the main focus of investigation, the recommendation here will only cover such aspects in the repository. As displayed in Figure 33, the UX page should contain the artifacts that are used in the UX work (Figure 33-B and 33-D). This refers to the repository page seen in Figures 29 and 30 previously presented. These are the pages/artifacts to which the US should be linked to. As showed in Figure 33, the arrangement of multiple UX Artifacts can be done within this same repository page. Considering this as an Epic UX page, all USs under such Epic would point to this same

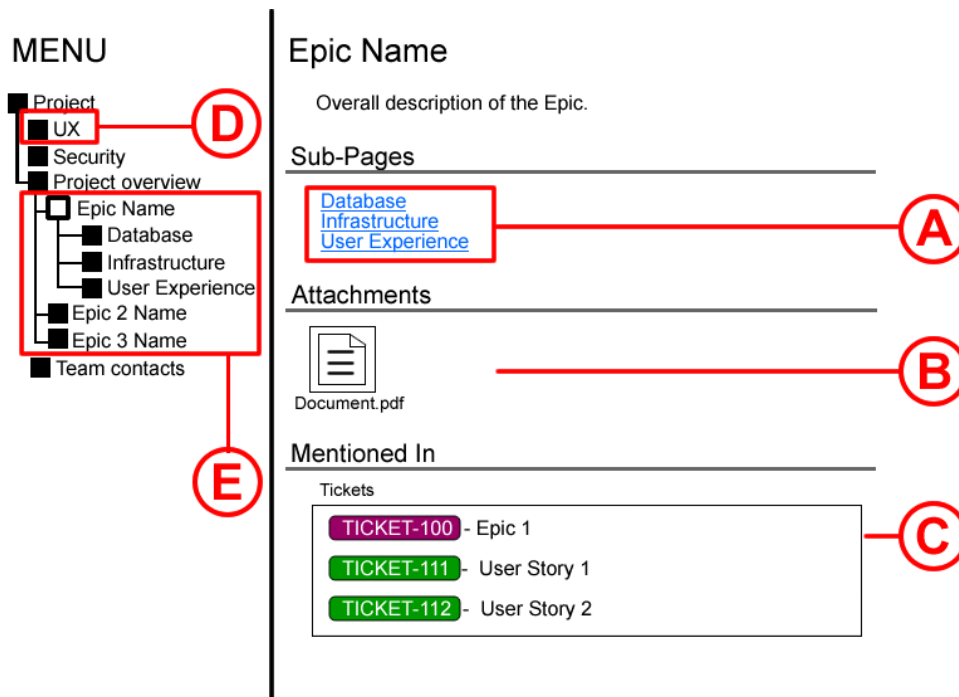


Figure 32: Specific Repository - Epic page

Source: Author

repository.

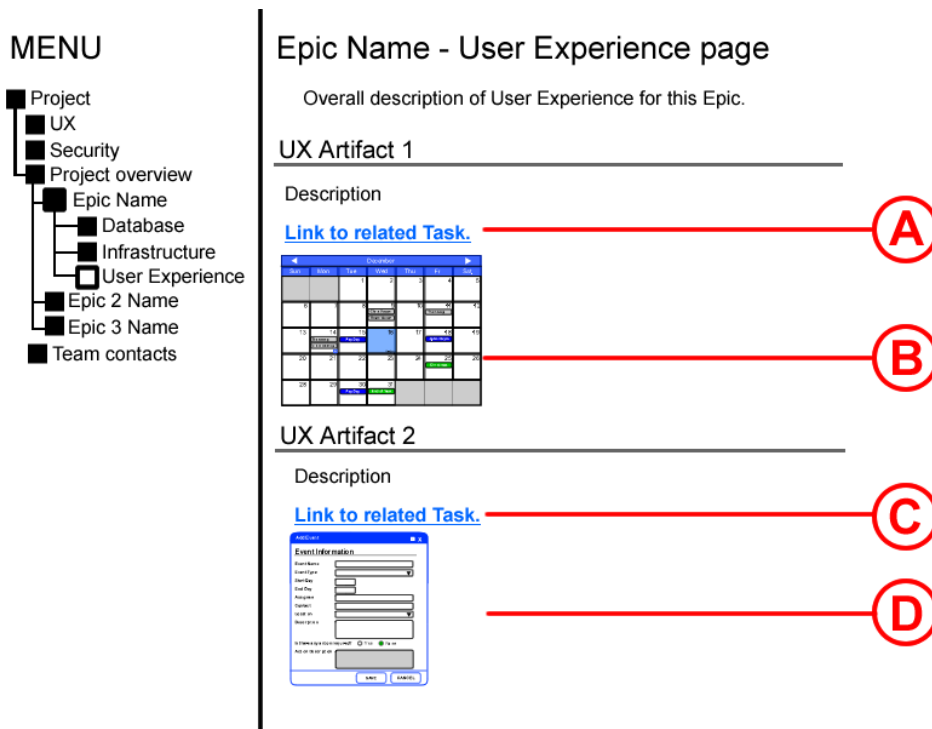


Figure 33: Specific Repository - UX page

Source: Author

Figure 33-A and 33-C represent a link to an US to where the such artifacts are related. Provided that such links are also placed in the US itself, it is possible to say that the navigational distance between the US and the UX information will be one. This means that the developer will only need to execute one navigational step to go from the US and the UX information. Depending on the artifact that is being placed in the central repository, more than one US may related to the same artifact. In such situations, the arrangement of the information into the central repositories becomes more evident, as the navigational distance from any US to the UX information will always be the same (provided the relation is created). Moreover, this is a solution to the information duplication problem that would occur in case the information was placed directly in the US level.

Finally, Figure 34 shows a sample of UX Central Repository, where UX information, that would be used across all the project, is placed. The following recommendation aims to create a place where "generic" UX information can be stored, without the need of duplication. Although the template presents a table with a set of information for each UX element, the creation of more information can be done. The main goal of the template is to have a central place for UX information used across the whole application. The amount of information available for each UX information is not covered in this study, although we believe that it can change to better satisfy each project needs.

For example, on Specific UX repositories, information related to an Epic and its USs would be stored. An example of such UX information could be the Mockup of a screen of a component that is particular to that screen. For the UX Central Repository though, examples of UX information that could be stored are button templates, that are to be used across the whole application. Another example, considering the usage of Personas as part of UX activities (PLONKA et al., 2014; ROGERS; SHARP; PREECE, 2019), therefore the storage of Personas in the UX Central repository could be done.

Moreover, such Central repository serves as a common knowledge place, helping in the sharing, and later finding, of UX information.














Icon	Font	Name	Description	
	Original	Btn_filter1	Button shown when a column is filtered. Btn_filter2's derivation	<b>Profit</b>  R\$ 1.500,00 R\$ 300,00 R\$ 100,00
	Original	Btn_filter2	Button shown when a column can be, but is not, filtered Btn_filter1's derivation	
	Bootstrap 2.3	Btn_plus	Button shown when a line can be expanded to show more information. Btn_minus's derivation	Name of investment  Invested value  Profit  Company A  R\$ 10.000,00 R\$ 1.500,00
	Bootstrap 2.3	Btn_minus	Button shown when a line can be retracted to show less information. Btn_plus Derivative	Company C R\$ 2.000,00 R\$ 100,00 Investment Apr/18 R\$ 1.000,00 R\$ 0 May/18 R\$ 500,00 R\$ 150,00 Jun/18 R\$ 500,00 R\$ 250
	Bootstrap 2.3	Btn_Home	Button symbolizing the home page of the application.	MY ACCOUNT ACTIONS MY INVESTMENTS  USER_123

Figure 34: UX Central Repository

Source: Author

### 4.4.3 Template usage in different Tools

The templates previously explained (subsections 4.4.2.1 to 4.4.2.3) can be applied in a variety of tools. On the other hand, the differences among the available tools, which encompass a variety of features may impact in the way the templates are created and managed. We briefly discussed in each of the previous subsections that the templates may be used in different tools, may it be a more complex tool, with a set of features that allow users to create and manage the templates in an easy way, or in basic/free tools, which may create some difficulties when trying to apply the templates.

Creating the templates in tools that contain more feature may help in the link management, especially if the tools have an automated integration. An example of such is when a link is create in one tool to another, another link is automatically created in the other tool. This helps by creating an environment with a "two-way" link, where you can navigate from one tool to another starting from any of the tools. Such feature may not be

available in all tools, making it difficult to create this "two-way" relationship, once that the same would need to be manually done and managed.

The templates also considered a set of information that may not fit all tools capabilities. Taking the US template as an example, when creating US in tools that don't allow a set of information as detailed as in the proposed US template, modifications to the template may be required. Such modifications were covered in the template, highlighting the minimum set of information that we believe that are essential for the template to still be valuable.

Although we recommend the usage of such templates in tools that allow the creation of similar structures to the ones presented in this study, the main features of each template have been exposed and, if the same are created any tool selected by the user, we believe that the templates, even with some modifications, can be applicable.

The main focus of the templates though, taking into consideration the US and UX repositories templates, is the UX information arrangement into the virtual environment (subsection 4.4.2.2). Provided that the templates used for the US and/or the UX repositories don't increase the navigational distance need to find given information and the final organization template is still the proposed one, we believe that the final navigational distance decrease (or limitation), as well as the navigational effort required to find UX information into the virtual environment, will remain the same as this proposal aims to reach. The final navigational classification would be a Wide navigational distance (Table 4), with limited navigational steps to find UX information from a given US, and a Medium navigational effort (Figure 24).

The template application into different tools and the impact of different tools into the templates is still to be further analyzed and is out of this study scope, although such verification is needed in order to understand when and how the template may be used into different tools.

#### 4.4.4 Recommendations considerations

Having Figure 23 as a diagram representation of the UX information dispersion within the virtual environments, using the proposed information placement here explained, we believe that a result similar to the presented in Figure 35 would be achieved.

Notice that in Figure 35, at "*Repository Tool*", the "*Epic Web Page*" refer to the specific UX Repository previously mentioned, being it related to a single Epic. The "*US-related information*" refers to the UX section particular to a US that is under the same Epic the repository relates to. UX central repository, containing the UX information to be used across the whole project (i.e. containing generic layouts/components) is represented by the "*Central UX Web Page*".

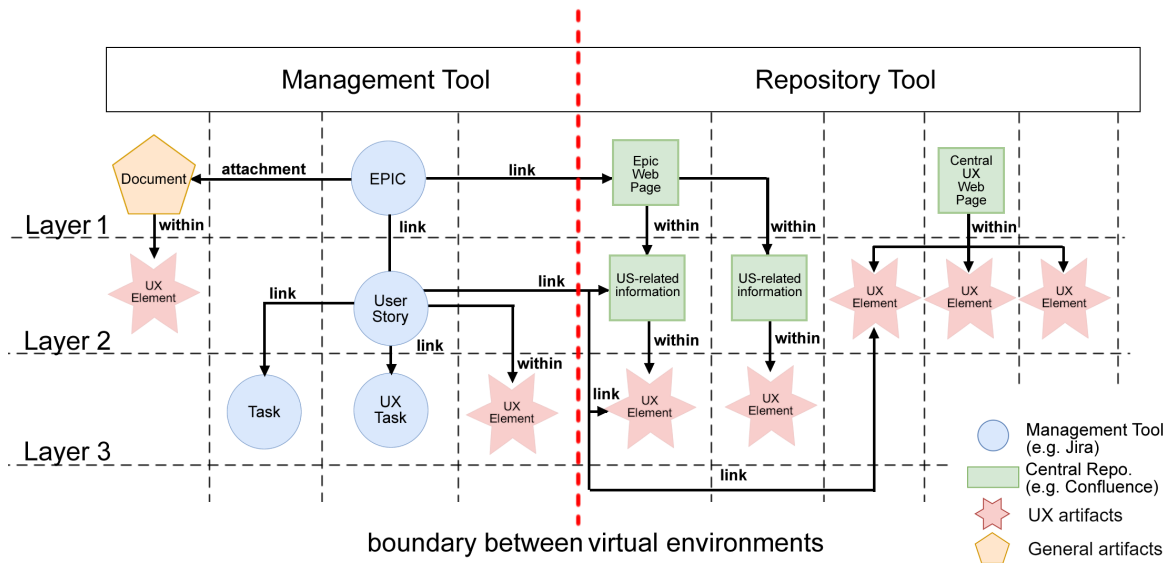


Figure 35: Distance diagram of proposed organization

Source: Author

The recommendations here presented covered a way to arrange the information into virtual environments, considering the navigation classification types and the navigation effort classification previously presented (Sections 4.2 and 4.3, respectively). Such classifications expand the work of Bjarnason et al. (2016), Bjarnason and Sharp (2017) and the recommendations aim to not only give insights to better arrange the information into virtual environments as to also try to create a pattern to reduce the navigational distances, having the case study (Chapter 3) navigational distances found as the basis for such improvement. We also considered some works to create an organization recommendation that would try to merge the UX work (PLONKA et al., 2014; ROGERS; SHARP; PREECE, 2019) with the actual software development work (SILVA et al., 2011; SILVA et al., 2018), which in agile practices usually has the US as the central point of attention (BIK; LUCASSEN; BRINKKEMPER, 2017).

The information organization template, although aiming to decrease the navigational distance in virtual environments, requires a minimal feature from the virtual tools being used. Its usage in basic tools may create problems on its usage, therefore we recommend its application in tools that provide a minimal tracking management. The application of the proposed templates is still to be evaluated and we recommend such evaluation to be performed in both basic and more complex tools in order to have a better understanding on when to apply such templates.

## 4.5 Final Considerations

In this chapter, we proposed a process to improve the navigational distances to UX information in virtual environments. Such recommendations have considered the work of [Bjarnason et al. \(2016\)](#), [Bjarnason and Sharp \(2017\)](#), regarding the navigational distance. The need to make UX closer to agile development, which has been reported by [Silva et al. \(2018\)](#). We also considered the US as the central focus of attention for the developers ([BIK; LUCASSEN; BRINKKEMPER, 2017](#); [COHN, 2004](#)). The proposed recommendations have the US as the starting point for the navigation process in the virtual environments.

We presented a comparison between practice and theory, by pointing issues that are reported in the literature on regards to UX and agile integration, and the outcomes of the case study earlier presented in this master work. Such comparison also had a role in providing insights to the development of the recommendations to reduce the navigational distances on UX elements. The recommendations have also considered some issues such as the UX elements traceability ([LEE; GUADAGNO; JIA, 2003](#); [KASHFI; NILSSON; FELDT, 2017](#)) importance and its dispersion in the virtual environments ([DESHPANDE et al., 2016](#); [SHARP; ROBINSON, 2006](#)).

The classification of the navigational distances expanded the work done by [Bjarnason et al. \(2016\)](#), [Bjarnason and Sharp \(2017\)](#). Furthermore, we also added a sub-category to the distances, considering the effort required for the navigation to be performed into virtual environments.

Based on the proposed navigation and effort categories, as well as the issues previously presented, we provided some good practice recommendations to reduce the navigational distances to UX information. Later, we presented a template proposal for the information arrangement in the virtual environments, having navigational distance improvement as the main goal.

By following the organization here proposed, the distances on the project would be of type Deep/Wide (i.e. no presence of Zero or Infinite distances, which are the extreme cases) with the effort being Medium, given the navigation through two different tools. Another point to be considered is the depth and width achieved in [Figure 35](#) arrangement.

The maximum depth of the information has been set to a maximum of three, considering that the developer will have to go to the Epic, US and one of its tasks to fully understand its context. If such is not necessary, the depth distance would be two, in case there is the need to go from the US to a task to get information. Considering the US information enrichment process described before, the arrangement Epic > US > Task could be taken as formal, but navigation through such hierarchy is not necessary all the time, provided that most of relevant information should be put in the US itself, reducing the navigational distance.

On the other hand, the arrangement of information into the repository, creating a relation to the US, makes the Wide distance to increase. Provided that a repository is created according to the Epics, and the US are linked to such repository, the maximum distance, considering the width, is also reduced. Limiting the distance to a repository, which should be the final destination to retrieve information, helps on decreasing the navigational distance. Figure 35 has a width distance of one, being it the navigation from the US to a Central Repository page, or from an Epic to the Epic Central Repository page.

Taking virtual environments as the main goal of such recommendations, the usage of USs as the central point for the information distribution was based on the fact that it is one of the most used artifacts into agile projects (BIK; LUCASSEN; BRINKKEMPER, 2017) and that it is a key project-related artifact used for tracking purposes (DESHPANDE et al., 2016). On the work of Deshpande et al. (2016), virtual artifacts and their supporting tools are stated as key information hubs (i.e. central places where information flows and decisions are made) for all team members, especially for remote workers, given these virtual artifacts dominate and shape their situation awareness<sup>2</sup>, and they are the core of their horizon of observation<sup>3</sup>.

Decreasing the navigational distance improves the information hub mentioned by Deshpande et al. (2016), once that it is easier to find the information. Creating central repositories can also be taken as buffers<sup>4</sup> if a team (e.g. UX team) is working in something that will not necessarily be used now (LDUF), placing such work in a specific central repository (i.e. UX repository), will not affect other information flow. Placing such information in the repository when these are to be used even if only in the future, will make it easier to find the information by them, once that the repository will be specific for such information.

The proposed information arrangement also aimed to keep the User Story as the main source of information for the teams. Following what was already concluded by Sharp and Robinson (2006), which states that, from the developer's point of view, the main transformation taking place in a cognitive system is transforming the US into executable code. In such work, there was little information propagation outside the US, being such the central focus of development from the time it is created until the code is handed over. The same principles are also supported by Cohn (2004), Sharp and Robinson (2006), Bik, Lucassen and Brinkkemper (2017), and the proposed information relation had such already known US importance in mind during its conception.

---

<sup>2</sup> Situation awareness: how people are kept informed of what is going on, e.g. through what they can see, what they can hear and what is accessible to them (DESHPANDE et al., 2016)

<sup>3</sup> Horizon of observation: what an individual can see or hear, influencing situation awareness (DESHPANDE et al., 2016).

<sup>4</sup> Buffers: location where information is held until it can be processed without causing disruption to ongoing activity (DESHPANDE et al., 2016).

Moreover, the proposed arrangement could be a way of decreasing one of the main challenges faced while establishing the integration of Agile and User-Centered Design. Garcia, Silva and Silveira (2019) describes this challenge as the question on how to facilitate communication among the invariably distinct involved practitioners. The creation of specific central repositories for UX information storage, letting it under UX practitioners management, while still relating such repositories to other environments and artifacts throughout the project (i.e. the management tools, the USs, etc.) is what the proposed arrangement aims to help with. The proposed arrangement also supports the idea of artifact-mediated communication, which has been previously concluded by Brhel et al. (2015) and Garcia, Silva and Silveira (2019).

The creation of a central repository location, based on a US scope, relates to the community understanding reported by Garcia, Silva and Silveira (2019), which states that the team must be cross-functional and the designer should be part of the team. By giving UX practitioners the ownership of a central repository and making its progress through a task under a US (Figure 29), which is mainly used by developers (SHARP; ROBINSON, 2006), is indirectly making the team work together, while still allowing each of the project members to work on its own pace and in its virtual environments. The virtual environments will later be integrated with a navigational distance that is within manageable limits and effort (i.e. navigational distance with Wide distance type and Medium effort classification, Tables 4 and 5 respectively).

Although the proposed information dispersion fits the Wide navigational distance classification, such scenario is good when applied when considering one agile artifact, such as the US, and one type of information, for example, the UX elements. If considering all the types of information that may be part of a software development process (Database, Architecture, Security, Quality Assurance, etc.), and each of these are considered as a different type of artifact to be linked to the US, the proposed template considering the Wide navigational distance as the best solution may not be applicable anymore. Such scenario is out of this study scope, which only considered the UX information, but the template being applied to other information types or along with it, is still a process to be further analysed in the future.

The templates proposed for the US writing considers a tool that enables users to be create a structure that contain a set of information that may not easily fit some tools, given that the US structure they allow is intended to be kept as small as possible (i.e. a card into Trello<sup>5</sup> tool). Therefore we encourage the usage of tools that do allow the creation of USs similar to the one proposed into the template.

Along with the US template, the template for the UX repositories considered a wiki-based tool. Depending on the tool used, the creation of similar structures may be more

---

<sup>5</sup> Trello: <https://trello.com>

difficult to do. Different from the US template, the UX repositories though intend to hold UX information and centralize it into a single location in the virtual environment. Moreover, given that wiki-based tools provide a freedom in the pages creation, the templates exposed in this study can vary, as long as its main principle (to centralize UX information) is kept and that the pages structure in such tool do not increase the navigational distance (i.e. keep information in only one location, being such related to an Epic).

Regarding the creation of links between tools, this study does not consider the difficulty on creating the same, once that depending on the tools used, the links may be automatically created and the tool may provide a quoting/mentioning system if the tools have an integration between them. Depending on the tools, such process may require more manual intervention, which difficulties the link creation when compared to tools that are integrated. The templates for the information organization into virtual environments only considered the creation of the link, without considering the effort to create such link. The main goal of the proposal is to enforce the link creation in order to reduce the final navigational distance between information, not to consider the effort required to create/keep such links.





## 5 Conclusion

This master project had as its main focus, the understanding of how UX elements and USs are linked in virtual environments. Next, the master project aimed to provide recommendations, which involved good practices and templates of information arrangement in virtual environments. Such recommendations had the goal to help on decreasing the navigational distance between UX information and agile artifacts, having the US as the starting point for the navigational distance calculation.

To achieve such recommendations and template proposals, the master project was divided into three main parts: An analysis of the literature, an industry case study and the recommendation creation process itself, which would be based on the two initial steps.

The literature analysis (Chapter 2) covered the main fundamentals used in this master project and the literature review. The literature review was designed based on issues reported in the literature was based on UX and agile practices integration. It also had a focus on the artifacts that are mainly used in agile practices and by UX specialists, how they relate in virtual environments and we also focused on the navigational distance (BJARNASON *et al.*, 2016; BJARNASON; SHARP, 2017). We also focused on the main issues being reported in the literature given the attempts of integrating UX-agile that have been done so far.

From the literature analysis, many issues have been found in integrating UX and Agile practices. Among many, artifact management, communication issues among team members and loss of big picture are the ones that can be highlighted. Considering such issues, the case study has been designed, aiming to understand, in an industry scenario, how UX information can be related to USs.

Next, this master project study presented an industry case study (Chapter 3), based on how UX information can be related to USs in a practical context. Furthermore, it analyzed the navigational distances present within virtual environments used by three projects in the industry. The analysis was designed based on issues reported in the literature on regards to UX and agile practices integration.

The case study was be divided in four main steps: An analysis over software requirement documents, to filter it and find the ones that had UX elements, having the work of Garrett (2010) to find such UX elements. The second step constituted of coding technique, based on the UX Elements framework proposed by Garrett (2010); on the software requirement documents that had been filtered in the first step. The third step was also a coding process but done over the USs, as well as other artifacts/virtual environments that had links from the USs, that originated from the software requirement documents

analyzed in step 2. The last step was an analysis of the dispersion of the UX elements coded in steps 2 and 3, to understand how they were arranged in the virtual environment used in the projects analyzed in the case study. Such UX information dispersion would later lead to insights on the UX elements navigational distance, which was based on the work of Bjarnason et al. (2016) and Bjarnason and Sharp (2017), which described the distances present in project development, including the one called navigational distance. An analysis of the distances between UX information and other artifacts used in the projects, being the US the starting point for the distance evaluation, was performed.

The navigational distances analysis concluded that there are no big distances in the virtual environments, i.e. the number of navigational steps to go from a US to the final UX information related to the US are within acceptable limits. Although such conclusion, the analysis showed that the UX information was still spread in the virtual environments. Given this problem, and considering the issues found in the literature review, recommendations on how to better place the UX information and relate it to USs in virtual environments. Furthermore, we also proposed categories for both navigational distances and the effort required to perform the navigation.

The improvement of navigational distances (Chapter 4) is the third and final main part of this master project. In this chapter we compared the findings from the case study with the issues reported in the literature. Having the outcomes of the literature review and the findings of the case study, good practice recommendations had been outlined in order to improve the awareness of agile practitioners on how UX information is important and what is the importance of having the navigational distance under manageable limits.

We also propose a classification for the navigational distance and the navigational effort, expanding the work of Bjarnason et al. (2016), Bjarnason and Sharp (2017), by adding a set of possible categories in which a given arrangement of UX information into virtual environments. Such categories contain pros and cons, which we outlined in the categories definition.

Having the issues reported in the literature, the outcomes from the case study and the navigational distance and effort categories, we proposed recommendations for good practices of UX information arrangement, which serve as an overview for the later proposed templates for the information arrangement into virtual environments.

The templates embrace a US template, the arrangement of UX information into virtual environments, having the US as a central point of attention. We also recommend templates for the creation of two types of UX repositories, the Specific repository, which holds UX information related to a limited scope (i.e. an Epic), and the Central repository, where "generic" UX information (i.e. used throughout the whole project, and therefore used in many Epics and USs) is stored.

The recommendations and templates aim to reduce the navigational distance between UX elements and USs into virtual environments.

Regarding the RQs outlined in the beginning of this master project: On RQ1 - *"How are UX information and USs connected into software virtual environments?"*, as exposed in Subsection 3.2.5.2, we can conclude that the UX information can be found in, at least, three different ways, being these: (i) Within: When information is in the body of the USs, i.e. within it; (ii) Link: When the information is arranged outside the US but still linked to it (through a hyperlink for example); (iii) Attachment: When the information is attached to the US.

Moreover, we can notice that information is not always present in the US itself, being placed in other locations. In such scenarios, the presence of navigational distance is more evident. Considering this, regarding RQ2 - *"What are the navigational distances found to access UX information from USs into software virtual environments?"*, we were able to classify it into four types: (i) Zero: When there is no need for navigation; (ii) Infinite: When the navigation is impossible to happen, unless by previous knowledge of the developer; (iii) Deep: Considering one tool, the deep distance refers to the need to go into more detailed (hierarchy) level into the same tool (as an example, going from the Epic to the US and then to the Task); (iv) Wide: When navigation happens through two or more tools, or within the same hierarchy level of a given tool (i.e. from a US to another US). Such classification have been outlined in Subsection 4.2.

Moreover, we introduced a new classification, considering the effort required to execute the navigation into virtual environments. Such classification was exposed in subsection 4.3, being summarized as: (i) Weak: When there is no need to navigate between different virtual environments; (ii) Medium: When there is the need to navigate between two virtual environments; (iii) Strong: When there is the need to navigate among three or more virtual environments; (iv) Broken: when the navigation through two or more virtual environments need to happen but is not possible due to poor linkage (i.e. link is not working anymore or is pointing to an invalid location, etc.). Such classifications answered the RQ2 and also provided an extension to the work of Bjarnason et al. (2016), Bjarnason and Sharp (2017), on regards to the navigational distance.

Below we introduce the contributions of this master project, as well as its limitations and possible future work.

## 5.1 Contributions

Some of the UX information mapped (coded) could be found within different contexts than the ones proposed by Garrett (2010). The initial insights led to the conclusion that, given agile practices are used in the projects, the separation among the planes proposed

by [Garrett \(2010\)](#) cannot be used in its totality. Despite the fact that each plane from the framework (Figure 4) is dependent of the others, in an agile project, it is possible to suggest that the planes are more tied to each other when referring to the time in which the information each of them holds are used in the project.

Although specific and distinct phases in the project life cycle may exist in an agile project, the fast delivery aimed by the practice pushes that all the information (represented by the planes in the framework) should be considered during most of the project life cycle and not in specific time frames within the same. The information, divided into the five UX elements by [Garrett \(2010\)](#), should be considered, in an agile scenario, within the same lead time (the time a process or development takes from its initiation to its release). For example, the surface aspects of a given development may be considered even in the beginning of its discussion. This makes the time-lapse among each plane to be very small, if even existent. This may be a hint for the reason that, during the coding (see Figure 12), artifacts and UX information could be placed in different planes than the ones mentioned in the work of [Garrett \(2010\)](#).

The second conclusion is regarding the usage of artifacts. Although some of the artifacts used in the projects matched the expected, given the literature review, others have not. As a highlight, it is possible to mention the Persona and Scenario artifacts usage, which are mentioned as one of the most used in the literature review but could not be directly found in the analysis. On the other hand, the Mockup artifact was used in all three projects analyzed. It is important to notice that none of the projects had a UX specialist, so the usage or not usage of some artifacts may be due to such team configuration.

The third conclusion is that, despite most information coded, present in documents, could also be found in US level, not all information could be mapped in USs itself. Information was found spread in other places, outside the US. Although a kind of relation between the information and the US was present, the initial thought, considering the study of [Bjarnason et al. \(2016\)](#) and the descriptions of navigational distance, was that the disposition of the UX information was not optimized.

Another point is that, although some information could be found in US level, it is valid to mention that one requirement document may generate many USs. Where each US only includes small pieces of work. This is reinforced by the coding performed in the virtual environment used by the teams, where complementary information could be found outside the USs, being placed in other virtual tools.

Such information disposition caused the presence of navigational distance. The analysis on the navigational distance didn't show it to be very large (i.e. it was not required many navigational steps to go from a US and the complementary UX information present outside of it). Given some of the issues already presented in the literature when merging agile and UX, such as artifact management, communication issues among team members

and loss of big picture, plus the fact that artifacts may be used as a communication mediator, the recommendations present in this study aimed to create a template of good practices to be followed when creating and relating UX information with the project's USs, which are mainly used by the project developers. Such recommendations were based in both navigational distance, graph-theory principles and US writing recommendations already present in the literature, having the purpose to help on mitigating the issue already mentioned, having the developers point of view as the base point for the recommendation application.

Apart from the recommendations, the study also proposed a new way to categorize the navigational distances found in the projects (Table 4) as well as the effort needed to execute such navigation (Table 5). These categories are a possible contribution to the work being done over the type of distances encountered in the projects, being such categorizations specifically focused on the distance called navigational distance.

## 5.2 Results Propagation

Regarding the project results propagation, apart from the final text here exposed, the creation of articles was also made.

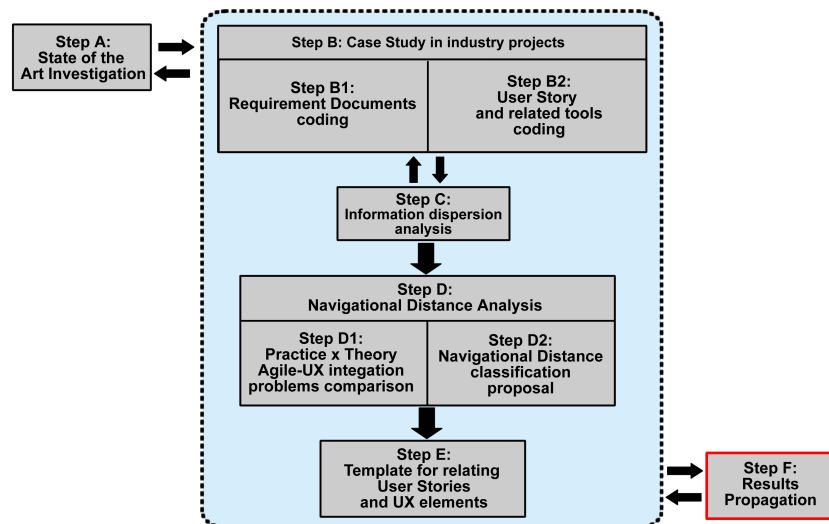


Figure 36: Results propagation

Source: Author

During the development of this project, an article entitled "Navigational distances between UX information and User Stories in agile virtual environments" was accepted on ICEIS 2020. We also intend to publish articles covering the UX information placement templates here exposed. After performing a validation of the proposed templates we also aim to publish another article covering the outcomes of the validation.

### 5.3 Study Limitations

The present master project had an industry case study evaluating which UX information could be found related to US and how the same was distributed into the virtual environments.

All projects analyzed used agile practices into non-agile environment (GREGORY *et al.*, 2016). No project had UX specialists in its team. Studying projects that do contain UX specialists could lead to new UX information being discovered in the coding phase, which would impact in the later analysis of the UX information propagation into different places.

The tools used by the projects were the same (i.e. Jira, Confluence and SharePoint). Although such tools are widely used (KASSAB, 2014; BERMAN; BARNETT; MOONEY, 2012), there are other similar tools that could also be analyzed. The multiple platforms cross-referencing could change from tool to tool, especially considering that some of them have automated processes that facilitate relationship creation and maintenance. Studying the same principles here presented in different tools could lead to new insights that were not covered in this study.

Although presenting some conclusions and recommending ways to better relate and arrange the UX information into virtual environments, such recommendation and templates were not verified into a real scenario. The recommendations, therefore, lack tests to confirm its efficiency. The confirmation of the impact of such recommendations is still to be evaluated.

The study only focused on relating UX to agile artifacts but in a real scenario, other information types should be considered. Relating UX with all the other information aspects that are included in a software development may impact in the conclusions presented here on regards to the proposed information organization templates.

The templates proposed are still to be validated. The generalization of such templates, being used into different virtual environments and with different tools is something not covered in this study, although we recommend the usage of tools that provide a set of features to track information and enable users to create a similar structure on the ones exposed in the templates. The generalization of such templates in tools that do not allow, or difficult, the creation of such templates is not covered in this study and is a topic to be later studied in order to better understand when such templates are feasible.

Considering other information that not only UX, the difference between the classified wide and deep distances may decrease, but the impact of multiple types of information and artifacts, that not only within the UX spectrum, being used in the proposed templates is still to be studied. Despite proposing templates for information organization, the templates would need a minimal feature being delivered by the tools used, otherwise all storage and

relationships being done in the virtual environments could be hard to be done, given the tool limitation. The application of the templates considering different tools is still to be evaluated.

## 5.4 Future Work

Further analysis of information being spread outside the virtual environment (i.e. in the real world) could be done to understand how UX information also propagates into the real world. Similar principles presented in this study could be applied, although the navigational distances would need to be revisited.

The current study also had navigational distance as a base for its analysis. The studies that proposed such distance (BJARNASON et al., 2016; BJARNASON; SHARP, 2017) also present other distances that could be further analyzed into the same context of UX information being used in agile projects.

To have a group study by using the UX information relationship, maintenance and disposition into virtual environments recommendations, evaluating the impact of such recommendations and how difficult the same are to be applied. Pros and Cons of such recommendations could be analyzed and lead to more refined recommendations.

Replicate the study process (coding, navigational distance analysis, etc.) into other agile projects, especially projects that contain UX specialists in its team, as well as projects that make usage of a different set of virtual tools could also bring new insights to the conclusions here presented.

Despite briefly analyzing the navigational distances into virtual environments, and even adding new categories to the same, contributing to the original navigational distances work (BJARNASON et al., 2016; BJARNASON; SHARP, 2017), analyzing the impact of the distances in the real world is also a topic to be better explored. Having such classifications to also evaluate the users (mainly project developers) emotions could also lead to the creation of new categories, as well as adding more description and/or weight to the same.





# Bibliography

- BECK, K. et al. *Manifesto for Agile Software Development*. 2001. <<http://agilemanifesto.org/>>. Online; accessed 23 September 2018. Mentioned 3 times at pages 21, 30, and 78.
- BERMAN, A.; BARNETT, W.; MOONEY, S. Collaborative software for traditional and translational research. *Human genomics*, v. 6, p. 21, 09 2012. Mentioned 2 times at pages 34 and 116.
- BIK, N.; LUCASSEN, G.; BRINKKEMPER, S. A reference method for user story requirements in agile systems development. In: IEEE. *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*. [S.l.], 2017. p. 292–298. Mentioned 6 times at pages 22, 31, 33, 105, 106, and 107.
- BJARNASON, E.; SHARP, H. The role of distances in requirements communication: a case study. *Requirements Engineering*, Springer, v. 22, n. 1, p. 1–26, 2017. Mentioned 23 times at pages 23, 24, 26, 27, 39, 45, 46, 47, 70, 71, 73, 75, 76, 79, 87, 89, 93, 105, 106, 111, 112, 113, and 117.
- BJARNASON, E.; SHARP, H.; REGNELL, B. Improving requirements-test alignment by prescribing practices that mitigate communication gaps. *Empirical Software Engineering*, Springer, v. 24, n. 4, p. 2364–2409, 2019. Mentioned in page 21.
- BJARNASON, E. et al. A theory of distances in software engineering. *Information and Software Technology*, Elsevier, v. 70, p. 204–219, 2016. Mentioned 23 times at pages 11, 23, 24, 26, 27, 38, 39, 45, 46, 70, 71, 75, 76, 79, 87, 93, 105, 106, 111, 112, 113, 114, and 117.
- BJARNASON, E.; WNUK, K.; REGNELL, B. A case study on benefits and side-effects of agile practices in large-scale requirements engineering. In: *Proceedings of the 1st Workshop on Agile Requirements Engineering*. [S.l.: s.n.], 2011. p. 1–5. Mentioned in page 30.
- BOHM, A. 5.13 theoretical coding: Text analysis in grounded theory. *A companion to qualitative research*, SAGE Publications, London, p. 270–275, 2004. Mentioned in page 53.
- BRHEL, M. et al. Exploring principles of user-centered agile software development: A literature review. *Information and Software Technology*, Elsevier, v. 61, p. 163–181, 2015. Mentioned 12 times at pages 21, 22, 39, 40, 42, 43, 46, 47, 77, 92, 93, and 108.
- BUDWIG, M.; JEONG, S.; KELKAR, K. When user experience met agile: a case study. In: ACM. *CHI'09 Extended Abstracts on Human Factors in Computing Systems*. [S.l.], 2009. p. 3075–3084. Mentioned 4 times at pages 22, 39, 40, and 47.
- CHOMA, J.; ZAINA, L. A.; BERALDO, D. Userx story: incorporating ux aspects into user stories elaboration. In: SPRINGER. *International Conference on Human-Computer Interaction*. [S.l.], 2016. p. 131–140. Mentioned 7 times at pages 22, 29, 31, 44, 47, 70, and 93.

- COHN, M. *User stories applied: For agile software development*. [S.l.]: Addison-Wesley Professional, 2004. Mentioned 18 times at pages 30, 31, 32, 44, 50, 54, 68, 77, 79, 88, 89, 90, 91, 93, 94, 96, 106, and 107.
- DESHPANDE, A. et al. Remote working and collaboration in agile teams. 2016. Mentioned 14 times at pages 23, 32, 33, 44, 45, 46, 76, 79, 80, 87, 88, 89, 106, and 107.
- Don Norman and Jakob Nielsen. *The Definition of User Experience*. 2013. <<http://www.nngroup.com/articles/definition-user-experience/>>. Online; accessed 22 March 2018. Mentioned 2 times at pages 21 and 34.
- DYBÅ, T.; DINGSØYR, T. Empirical studies of agile software development: A systematic review. *Information and software technology*, Elsevier, v. 50, n. 9-10, p. 833–859, 2008. Mentioned 3 times at pages 30, 78, and 79.
- ERGONOMICS of human-system interaction – Part 210: Human-centered design for interactive systems. [S.l.], 2010. v. 2000. Mentioned in page 34.
- GARCIA, A.; SILVA, T. S. da; SILVEIRA, M. S. Artifact-facilitated communication in agile user-centered design. In: SPRINGER. *International Conference on Agile Software Development*. [S.l.], 2019. p. 102–118. Mentioned 9 times at pages 22, 35, 39, 43, 44, 77, 92, 93, and 108.
- GARCIA, A.; SILVA, T. Silva da; SILVEIRA, M. S. Artifacts for agile user-centered design: A systematic mapping. In: *Proceedings of the 50th Hawaii International Conference on System Sciences*. [S.l.: s.n.], 2017. Mentioned 12 times at pages 22, 35, 39, 40, 42, 44, 46, 47, 76, 77, 92, and 93.
- GARRET, J. J. *The Elements of User Experience*. 2000. <<http://www.jjg.net/elements/pdf/elements.pdf>>. Online; accessed 20 December 2018. Mentioned in page 36.
- GARRETT, J. J. *Elements of user experience, the: user-centered design for the web and beyond*. [S.l.]: Pearson Education, 2010. Mentioned 19 times at pages 24, 25, 31, 34, 35, 36, 37, 38, 40, 46, 47, 53, 56, 61, 64, 65, 111, 113, and 114.
- GREGORY, P. et al. The challenges that challenge: Engaging with agile practitioners' concerns. *Information and Software Technology*, Elsevier, v. 77, p. 92–104, 2016. Mentioned 2 times at pages 50 and 116.
- HARRIS, J. M.; HIRST, J. L.; MOSSINGHOFF, M. J. *Combinatorics and graph theory*. [S.l.]: Springer, 2008. Mentioned 3 times at pages 71, 74, and 83.
- HASSENZAHN, M.; TRACTINSKY, N. User experience-a research agenda. *Behaviour & information technology*, Taylor & Francis, v. 25, n. 2, p. 91–97, 2006. Mentioned 2 times at pages 21 and 77.
- HESS, A.; DIEBOLD, P.; SEYFF, N. Towards requirements communication and documentation guidelines for agile teams. In: IEEE. *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*. [S.l.], 2017. p. 415–418. Mentioned 6 times at pages 21, 22, 39, 42, 43, and 47.
- INAYAT, I. et al. A systematic literature review on agile requirements engineering practices and challenges. *Computers in human behavior*, Elsevier, v. 51, p. 915–929, 2015. Mentioned 5 times at pages 31, 46, 76, 78, and 79.

JURCA, G.; HELLMANN, T. D.; MAURER, F. Integrating agile and user-centered design: a systematic mapping and review of evaluation and validation studies of agile-ux. In: IEEE. *2014 Agile Conference (AGILE)*. [S.l.], 2014. p. 24–32. Mentioned 8 times at pages 21, 39, 40, 47, 76, 77, 78, and 88.

KASHFI, P.; NILSSON, A.; FELDT, R. Integrating user experience practices into software development processes: implications of the ux characteristics. *PeerJ Computer Science*, v. 3:e130, 2017. Mentioned 6 times at pages 43, 46, 76, 77, 92, and 106.

KASSAB, M. An empirical study on the requirements engineering practices for agile software development. In: IEEE. *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*. [S.l.], 2014. p. 254–261. Mentioned 4 times at pages 32, 33, 43, and 116.

KUPIAINEN, E.; MÄNTYLÄ, M. V.; ITKONEN, J. Using metrics in agile and lean software development—a systematic literature review of industrial studies. *Information and Software Technology*, Elsevier, v. 62, p. 143–163, 2015. Mentioned in page 29.

LAW, E. L.-C. et al. Understanding, scoping and defining user experience: a survey approach. In: ACM. *Proceedings of the SIGCHI conference on human factors in computing systems*. [S.l.], 2009. p. 719–728. Mentioned in page 34.

LEE, C.; GUADAGNO, L.; JIA, X. An agile approach to capturing requirements and traceability. In: *Proceedings of the 2nd International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE 2003)*. [S.l.: s.n.], 2003. v. 20. Mentioned 9 times at pages 45, 46, 76, 77, 83, 87, 88, 90, and 106.

LEE, J. C.; MCCRICKARD, D. S. Towards extreme(ly) usable software: Exploring tensions between usability and agile software development. In: IEEE. [S.l.], 2007. p. 59–71. Mentioned in page 41.

LISKIN, O. How artifacts support and impede requirements communication. In: SPRINGER. *International Working Conference on Requirements Engineering: Foundation for Software Quality*. [S.l.], 2015. p. 132–147. Mentioned 14 times at pages 39, 41, 42, 43, 47, 62, 70, 73, 87, 88, 89, 90, 95, and 99.

LOPES, L. A. et al. Adding human interaction aspects in the writing of user stories: a perspective of software developers. In: ACM. *Proceedings of the 31st Brazilian Symposium on Software Engineering*. [S.l.], 2017. p. 194–203. Mentioned 3 times at pages 22, 40, and 47.

LUCASSEN, G. et al. Forging high-quality user stories: towards a discipline for agile requirements. In: IEEE. *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. Ottawa, ON, Canada, 2015. p. 126–135. Mentioned 8 times at pages 22, 87, 88, 89, 91, 93, 94, and 96.

MATHARU, G. S. et al. Empirical study of agile software development methodologies: A comparative analysis. *ACM SIGSOFT Software Engineering Notes*, ACM, v. 40, n. 1, p. 1–6, 2015. Mentioned 2 times at pages 30 and 78.

MEMMEL, T.; GUNDELSWEILER, F.; REITERER, H. Agile human-centered software engineering. In: BRITISH COMPUTER SOCIETY. *Proceedings of the 21st British HCI*

*Group Annual Conference on People and Computers: HCI... but not as we know it- Volume 1.* [S.l.], 2007. p. 167–175. Mentioned in page [40](#).

MORENO, A. M.; YAGÜE, A. Agile user stories enriched with usability. In: SPRINGER. *International Conference on Agile Software Development.* [S.l.], 2012. p. 168–176. Mentioned 2 times at pages [44](#) and [47](#).

PETERSEN, K.; WOHLIN, C. A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. *Journal of systems and software*, Elsevier, v. 82, n. 9, p. 1479–1490, 2009. Mentioned in page [21](#).

PLONKA, L. et al. Ux design in agile: a dsdm case study. In: SPRINGER. *International Conference on Agile Software Development.* [S.l.], 2014. p. 1–15. Mentioned 4 times at pages [22](#), [40](#), [102](#), and [105](#).

REED, A. H.; KNIGHT, L. V. Effect of a virtual project team environment on communication-related project risk. *International Journal of Project Management*, Elsevier, v. 28, n. 5, p. 422–427, 2010. Mentioned 2 times at pages [23](#) and [32](#).

ROGERS, Y.; SHARP, H.; PREECE, J. *Interaction Design: Beyond Human-Computer Interaction.* [S.l.]: Wiley Publishing, 2019. ISBN 9781119547259. Mentioned 7 times at pages [35](#), [36](#), [46](#), [76](#), [78](#), [102](#), and [105](#).

RUNESON, P.; HÖST, M. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, Springer, v. 14, n. 2, p. 131, 2009. Mentioned in page [51](#).

SCHÖN, E.-M.; THOMASCHEWSKI, J.; ESCALONA, M. J. Agile requirements engineering: A systematic literature review. *Computer Standards & Interfaces*, Elsevier, v. 49, p. 79–91, 2017. Mentioned in page [35](#).

SCHÖN, E.-M. et al. Key challenges in agile requirements engineering. In: SPRINGER. *International Conference on Agile Software Development.* [S.l.], 2017. p. 37–51. Mentioned 6 times at pages [39](#), [42](#), [43](#), [44](#), [47](#), and [91](#).

SCHÖN, E.-M. et al. Enterprise experience into the integration of human-centered design and kanban. In: *ICSOFT-EA.* [S.l.: s.n.], 2016. p. 133–140. Mentioned 2 times at pages [87](#) and [89](#).

SCHWARTZ, L. Agile-user experience design: an agile and user-centered process? In: *Proc. the 8th International Conference on Software Engineering Advances.* [S.l.: s.n.], 2013. p. 346–351. Mentioned in page [40](#).

SHARP, H.; ROBINSON, H. A distributed cognition account of mature xp teams. In: SPRINGER. *International Conference on Extreme Programming and Agile Processes in Software Engineering.* [S.l.], 2006. p. 1–10. Mentioned 7 times at pages [44](#), [46](#), [76](#), [79](#), [106](#), [107](#), and [108](#).

SHUKLA, V.; AURIOL, G.; BARON, C. A graph-based requirement traceability maintenance model. In: *Sixth International Conference on Software Engineering Advances.* [S.l.: s.n.], 2011. p. 161–165. Mentioned 3 times at pages [45](#), [87](#), and [89](#).

- SILVA, T. S. D. et al. User-centered design and agile methods: a systematic review. In: IEEE. *Agile Conference (AGILE)*, 2011. [S.l.], 2011. p. 77–86. Mentioned 11 times at pages [22](#), [39](#), [40](#), [42](#), [46](#), [47](#), [76](#), [77](#), [92](#), [99](#), and [105](#).
- SILVA, T. S. D. et al. The evolution of agile uxd. *Information and Software Technology*, Elsevier, v. 102, p. 1–5, 2018. Mentioned 12 times at pages [40](#), [44](#), [76](#), [77](#), [87](#), [90](#), [91](#), [92](#), [93](#), [99](#), [105](#), and [106](#).
- SOARES, H. F. et al. Investigating the link between user stories and documentation debt on software projects. In: IEEE. *Information Technology-New Generations (ITNG), 2015 12th International Conference on*. [S.l.], 2015. p. 385–390. Mentioned 5 times at pages [22](#), [42](#), [87](#), [90](#), and [91](#).
- STOPA, G. R.; RACHID, C. L. Scrum: Metodologia ágil como ferramenta de gerenciamento de projetos. *CES Revista*, v. 33, n. 1, p. 302–323, 2019. Mentioned in page [33](#).
- STRAUSS, A.; CORBIN, J. M. *Basics of qualitative research: Grounded theory procedures and techniques*. [S.l.]: Sage Publications, Inc, 1990. Mentioned in page [25](#).
- TAIBI, D. et al. Comparing requirements decomposition within the scrum, scrum with kanban, xp, and banana development processes. In: SPRINGER. *International Conference on Agile Software Development*. [S.l.], 2017. p. 68–83. Mentioned 4 times at pages [21](#), [32](#), [79](#), and [96](#).
- VINSON, N. G. Design guidelines for landmarks to support navigation in virtual environments. In: ACM. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. [S.l.], 1999. p. 278–285. Mentioned in page [89](#).
- VREDENBURG, K. et al. A survey of user-centered design practice. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. [S.l.: s.n.], 2002. p. 471–478. Mentioned in page [34](#).
- WILSON, J.; ROSENBERG, D. Rapid prototyping for user interface design. In: *Handbook of human-computer interaction*. [S.l.]: Elsevier, 1988. p. 859–875. Mentioned in page [35](#).
- WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: ACM. *Proceedings of the 18th international conference on evaluation and assessment in software engineering*. London, England, United Kingdom, 2014. p. 38. Mentioned in page [39](#).



# APPENDIX A – Coding of Document 1

For this document, 30 use cases were analyzed, distributed in 6 different documents. From these, 21 use cases contained aspects related to interface aspects (UX). After the analysis, it was identified that such use cases had a documentation standard which is exposed below. Given data confidentiality, no real data will be presented here. Instead, documents have been written based on the documents analyzed. The documents presented contain similar structure to the ones analyzed, summarizing its main structures. Similar requirements and how they were written are also presented, omitting confidential data, being displayed a "generic" software requirement in its place.

# 1. Document Information

This document refers to the new XYZ functionality (s) which was requested by the ABC department to reduce the need to use non-computerized means to manage and share events that occur both in the department and in the company as a whole.

## 2. Finance's Diary

The Finance Diary feature will be a new feature to be implemented in the XYZ project, which has the purpose of facilitating the events management for the finance sector. Users will be able to navigate to the diary screen, located on the top menu, which will display a calendar containing all available events. When you click on a calendar item, a popup with event information will be displayed. It will also be possible, for certain positions, to create and edit events, functionality available on the same screen.

### 2.2 Use Case 1: Visualize Finance's Diary

#### 2.2.1 Description

The Finance Diary view will be displayed through the main screen by clicking the "Finance Diary" button (currently non-existent) located in the top menu of the screen. When you click on the menu, a new screen, Finance Diary, should be displayed containing a calendar with the current month and all events loaded in it.

- Feature includes a new screen (Finance's Diary)
- The visualization can be done through the "Finance diary" button from the top menu of the main screen.

#### 2.2.2 Actors

- Finances
- Directors

#### 2.2.3 Pre-Condition

- The user is within the allowed group for viewing

#### 2.3.4 Workflow

- a. User navigates to the main screen and clicks the "Finance Diary" button located on the top menu.

PS.: If the user does not have access permission, the button should not be displayed.

- b. Finance Diary screen is displayed.



December						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1	2	3	4	5
6	7	8	9 Chris Vacan. Thom Vacan.	10	11 Training.	12
13	14 Training. Client meeting +	15 Pay Day	16 today	17	18 John Paym.	19
20	21	22	23	24	25 Christmas	26
28	29	30 Pay Day	31 End of Year			

Figure 1 – Calendar View Mock

1. The user can navigate through the calendar months by clicking on the arrows located at the top
2. When clicking on an event, a Pop-Up should appear showing the event information.

3. Calendar events should be color-coded according to the following rules:
  - Gray: Events in the past
  - Green: Holiday Events
  - Blue: Events of type Payment
  - [...]

### 2.3.5 Post condition

N/A

## 2.2 Use Case 2: Add new Finance Diary Event

### 2.2.1 Description

Users will be able to add new events to the Finance Diary, which should appear in the Calendar's calendar view (see use case 1). Events can be added directly on the Diary screen or through a widget available on the main application screen.

- The functionality includes a new widget on the main screen
- Creating a new event can be done through the Diary screen or from the widget on the main screen

### 2.2.2 Actor

- Finance

### 2.2.3 Pre-Condition

- The user is within the allowed group for creating events

### 2.3.4 Workflow

1. User navigates to the main screen and clicks in the "Add Finance event" widget available on the page footer.

PS.: If the user does not have access permission, the button should not be displayed.

2. A modal with the fill form for creating a new event is displayed.

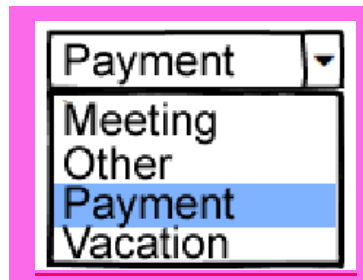
The image shows a mockup of a modal form titled "Add Event". The form is set against a light blue background with a white border. It contains the following elements:

- Event Information** (Section Header)
- Event Name:** A text input field.
- Event Type:** A dropdown menu with a downward arrow.
- Start Day:** A text input field.
- End Day:** A text input field.
- Assignee:** A text input field.
- Contact:** A text input field.
- Location:** A dropdown menu with a downward arrow.
- Description:** A text area.
- Is there any action required?:** Two radio buttons, "True" and "False". The "False" radio button is selected.
- Action Description:** A text area.
- SAVE** and **CANCEL** buttons at the bottom.

Figure 2 – Create Finance event pop-up form mock

- a. Event Name should be a text field that allows up to 50 characters.
- b. Event Type should be a combobox with the following options:
  - Vacation
  - Audit
  - Payment

- Receiving



[...]

PS.:

- All fields are required
- If creation is done through the Finance Diary screen, the calendar should be automatically updated with the new information

### 2.3.5 Post conditions

- Validate that the event was actually created and inform the user of its creation
- The event should be available for viewing on the Finance Diary screen, appearing on the corresponding day in the calendar view.

### 2.3.6 UX screen sample



Figure 3 – Finance Diary Widget for main screen



## APPENDIX B – Coding of Document 2

For this document, 18 use cases were analyzed, distributed in 4 different documents. Of these, 8 use cases contained aspects related to interface aspects (UX). The patterns found are exposed below. Given data confidentiality, no real data will be presented here. Given data confidentiality, no real data will be presented here. Instead, documents have been written based on the documents analyzed. The documents presented contain similar structure to the ones analyzed, summarizing its main structures. Similar requirements and how they were written are also presented, omitting confidential data, being displayed a "generic" software requirement in its place.

# 1. Document Information

This document pertains to the investment screen feature (s), which includes a new feature in the existing AAA application. This functionality includes the creation of a new screen that will provide information regarding the user's financial applications.

## 2. Use Case 1 – Investment screen

### 2.1 User Stories

#### 2.1.1 User Story 1 – My Investments Screen

Project Owner: John Titor

Related Jiras	Release Version	Release Date
<a href="#">PROJECT-111</a>	2018.1	30-03-2018
<a href="#">PROJECT-112</a>	2018.1	30-03-2018
<a href="#">PROJECT-113</a>	2018.2	30-06-2018
...	...	...

##### 2.1.1.1 Description

This User Story refers to the new "My Investments" screen, which is intended to enable the user to manage their financial investments. Use Case 1 is the creation of a new screen, through which the user can view all his current investments, grouping them by name of the investment showing both the name and the value currently invested, as well as the amount of profit (being highlighted in green) or loss (being highlighted in red). The screen should have options for viewing details of the investment as well as options for filtering the investments by name or by invested amount.

##### 2.1.1.2 Actors

- All users

##### 2.1.1.3 Acceptance Criteria

- Ensure that the screen is accessible through the menu "My Investments"
- Ensure that all investments are listed, initially in alphabetical order
- It should be possible to filter the investments and sort them by name or value
- Profits should be shown in green color
- Prejudices should be shown in red color
- Field to display application information should be available
- Validate investment information (details)
- Approval during UAT phase (user acceptance test)

##### 2.1.1.4 Dependencies

- Creating a new access menu (use case 3)
- Navigate to the screen via the start menu (user story 2)

### 3. Integration with external ABC application

#### 2.1.1.5 Story Details

Currently there is no screen to view all the investments, being necessary to access them one by one. This user story aims to meet this need by creating a new screen, from which it will be possible to have quick and easy access to all the investments as well as to have a summary of the main information of the same ones directly in this screen.

This new screen will be of exclusive access of each user according to their respective investments, having only functionality of visualization of the data.

[Screen mock.](#)

#### 2.1.1.6 Data Elements / Data Types

Field	Type	Condition	Description
Investment Name	String	Mandatory	Name of the investment
Invested Value	Long	Mandatory	Current invested amount
Profit	Long	Mandatory	Profit or loss obtained in the last month, being shown in green for profit and red for loss.
More Information	Button / Icon	Mandatory	Expands the component by displaying investment information.

## 2.2.1 User Story 2 – My Investments Menu

### 2.2.1.1 Description

This User Story refers to the new "My Investments" menu, which will be available in the top menu of the application, allowing the user to navigate to the new "My investments" screen (use case 1).

### 2.2.1.2 Actors

- All users

### 2.2.1.3 Acceptance Criteria

- Ensure that the new "My Investments" menu is located in the top menu.
- Ensure that the new menu is not a sub-item of an already existing menu.
- Ensure that the new menu is accessible from all the application screen.
- Ensure that the "My Investments" screen (use case 1) is accessible through the menu "My Investments"

### 2.2.1.4 Dependencies

- Creation of the new My Investments screen (use case 1)

### 2.2.1.5 Story Details

Currently there is no screen to view all the investments, being necessary to access them one by one. After creating such a screen (user story 1), creating a new menu (my investments) is necessary in order to allow navigation to the new screen (my investments). The new menu should be located in the top menu of the application and should not be sub-item of any already existing menu.

[Screen mock.](#)

### 2.2.1.6 Data Elements / Data Types

Field	Type	Condition	Description
My Investment Menu	Button	Mandatory	My Investment menu. Will link to My Investment screen.



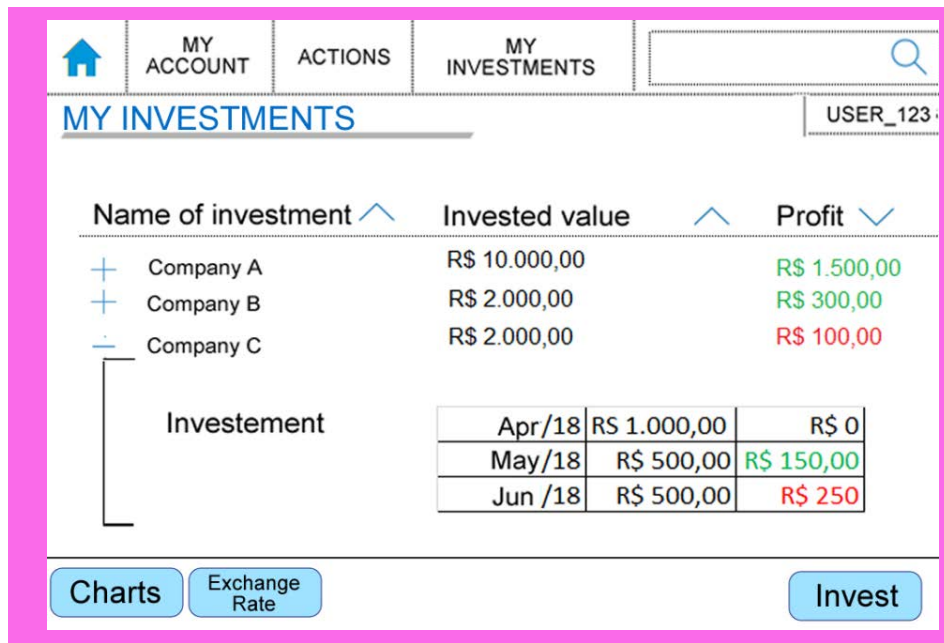
## 4. Interface Documentation

### 4.1 Application Flow

Home > My Investments

### 4.2 Interface 1 – My Investments

Full Screen mockup



#### 4.2.1 Summary

New screen, my investments, accessed through the initial menu. This screen will display all the user's investments, as well as the balance and main information of each investment.

#### 4.2.2 User Story

- US1
  - As an investor I want to see my investments so that I can manage my money.
- US1
  - As an investor I want to see my profit and loss so that I can be aware of my investment situation.
- US1
  - As an investor I want to see my investment details so that I can have a balance of the previous months.

#### 4.2.3 Description of Requirements

A design screen using gray and blue colors, according to the application standard. Titles should be described in bold. Expand / retract buttons should be placed on the left side of the screen, while the right side should now be covered with the data, although in the future there is the possibility of adding action buttons in this field.

#### 4.2.4 Gherkin Scenarios

Given an user has in investments	When the user wants to check all investments	Then go to investments screens
Given an user wants to see profit and loss for all investments	When user is in the investments screen	Then use identify profit as green while loss is red
Given an user is in the investment screen	When the user want to check investment details	Then click in the plus sign in on the right side of the investment

## 4.2.5 Mockups & Flows

### Investment Screen

Name of investment ^	Invested value ^	Profit v	
+ Company A	R\$ 10.000,00	R\$ 1.500,00	
+ Company B	R\$ 2.000,00	R\$ 300,00	
- Company C	R\$ 2.000,00	R\$ 100,00	
Investement	Apr/18	R\$ 1.000,00	R\$ 0
	May/18	R\$ 500,00	R\$ 150,00
	Jun /18	R\$ 500,00	R\$ 250

### 4.2.6 Security

This screen will only display data for viewing and only information regarding the user in question should be shown.

### 4.2.7 Q&A

## 4.3 Interface 2 – My Investments Menu

### 4.3.1 Summary

New menu, my investments, located in the initial menu. This menu will navigate to the My Investments screen.

### 4.3.2 User Story

- US2
  - **As an investor I want to see all my investments so that I can manage my money.**

### 4.3.3 Description of Requirements

New menu titled My Investments located in the top menu. From there you will navigate to the My Investments screen.

### 4.3.4 Gherkin Scenarios

Given an user has in investments	When the user wants to check all investments	Click on Meus Investimentos button
----------------------------------	--	------------------------------------

### 4.3.5 Mockups & Flows


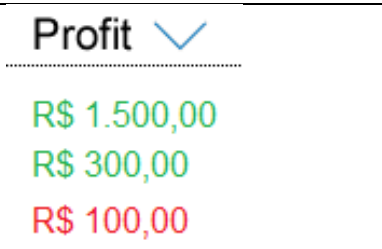


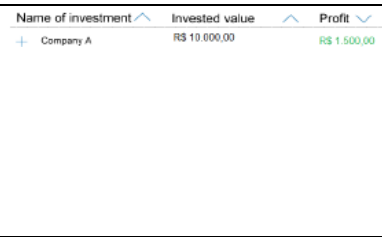

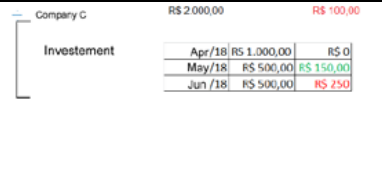






### 4.3.6 Security

N/A

### 4.3.7 Q&A

## 5. Frontend Documentation

Icon	Font	Name	Description	
	Original	Btn_filter1	Button shown when a column is filtered. Btn_filter2's derivation	
	Original	Btn_filter2	Button shown when a column can be, but is not, filtered Btn_filter1's derivation	
	Bootstrap 2.3	Btn_plus	Button shown when a line can be expanded to show more information. Btn_minus's derivation	
	Bootstrap 2.3	Btn_minus	Button shown when a line can be retracted to show less information. Btn_plus Derivative	
	Bootstrap 2.3	Btn_Home	Button symbolizing the home page of the application.	
	Bootstrap 2.3	Btn_magnifier	Magnifying glass icon. Used in every field related to searches.	



## APPENDIX C – Coding of Document 3

For this document, 20 use cases were analyzed, distributed in 3 different documents. Of these, 6 use cases contained aspects related to interface aspects (UX). The pattern are exposed below. Given data confidentiality, no real data will be presented here. Instead, documents have been written based on the documents analyzed. The documents presented contain similar structure to the ones analyzed, summarizing its main structures. Similar requirements and how they were written are also presented, omitting confidential data, being displayed a "generic" software requirement in its place.

## 1.0. Introduction

### 1.1. Purpose

This document refers to the "User screen" functionality that was requested by the Human Resources department, aiming at reducing the need to use non-computerized means to manage employee's information.

### 1.2. Scope of Project

This functionality is composed by the implementation of the "User screen" functionality. The functionality will include operations of visualization, addition, editing and removal of the data, as detailed in this document.

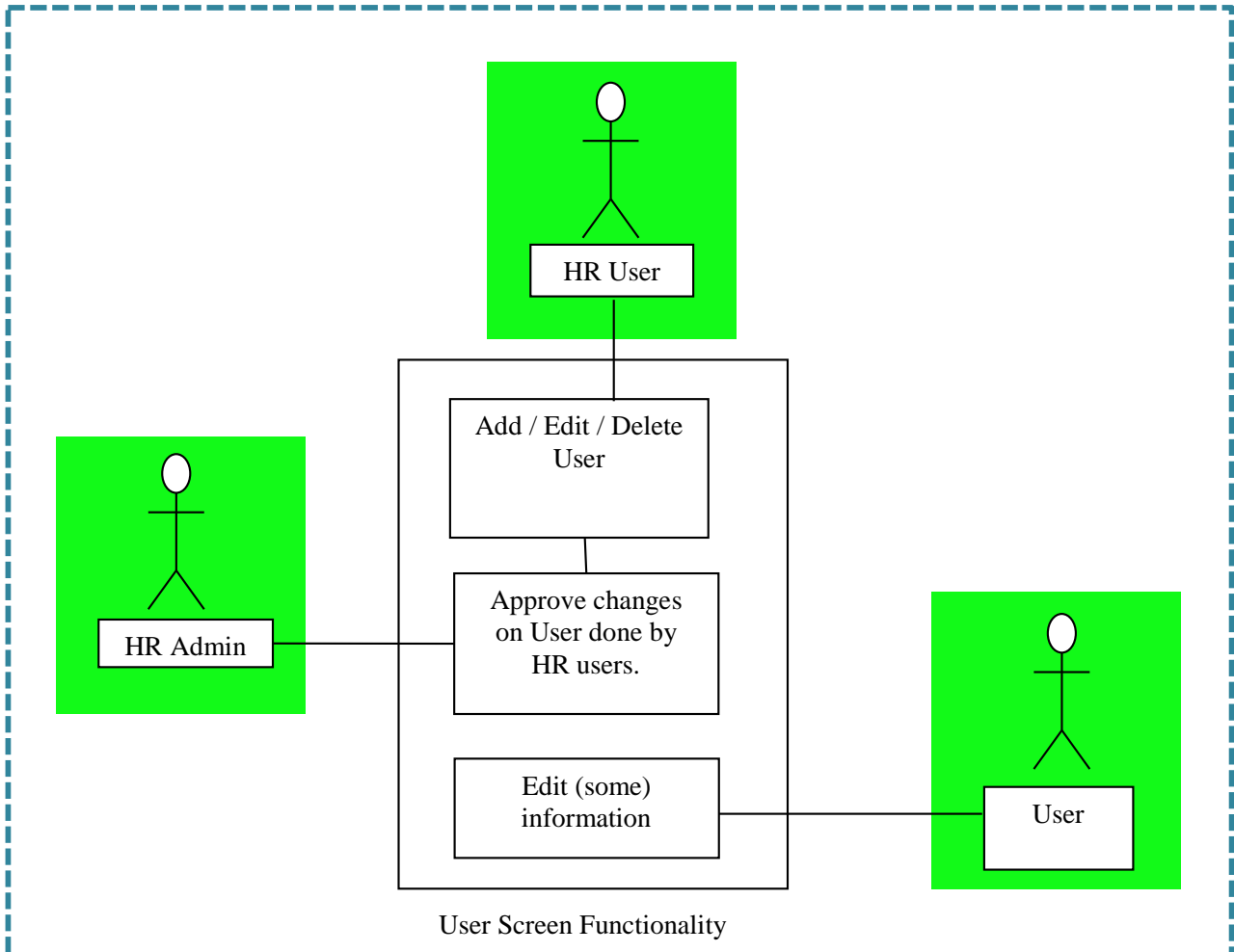
### 1.3. Glossary

<b>Term</b>	<b>Definition</b>
User	Reviewer or Author.



## 2.0. Overall Description

### 2.1 System Environment



**Figure 1 - System Environment**

The human resources sector contains four active actors who will operate the system. The actors will be classified into two groups: Administrator and HR User. Three actors will belong to the HR User group, having access to all the functionalities, however any approval must be made by the fourth actor, denoted as administrator.

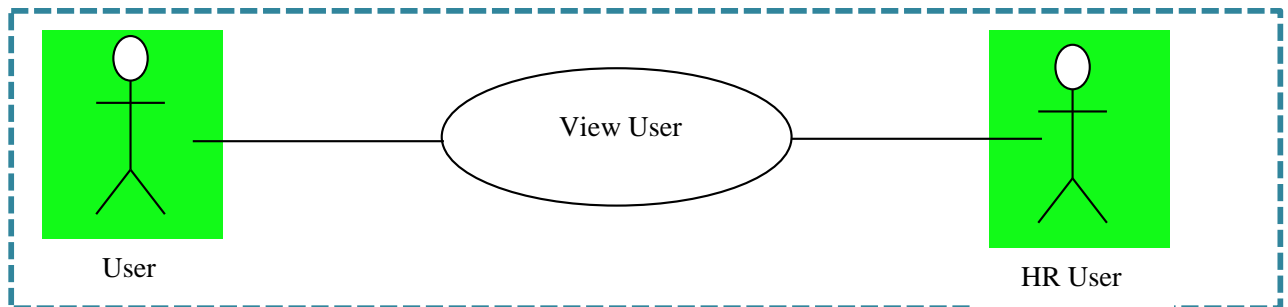
## 2.2 Functional Requirements Specification

This section explains the use cases for each of the actors involved. All involved will have the same use cases, however the Administrator user will have one more action, this being of approval.

### 2.2.1 Use Cases

#### 2.2.1.1: View User

Diagram:



#### Brief Description

The User view will be displayed through the main screen by clicking on the "Users" button (currently nonexistent) located in the top menu of the screen. When clicking on the menu, a new screen, Users, should be displayed containing user information, as well as a photo of the same.

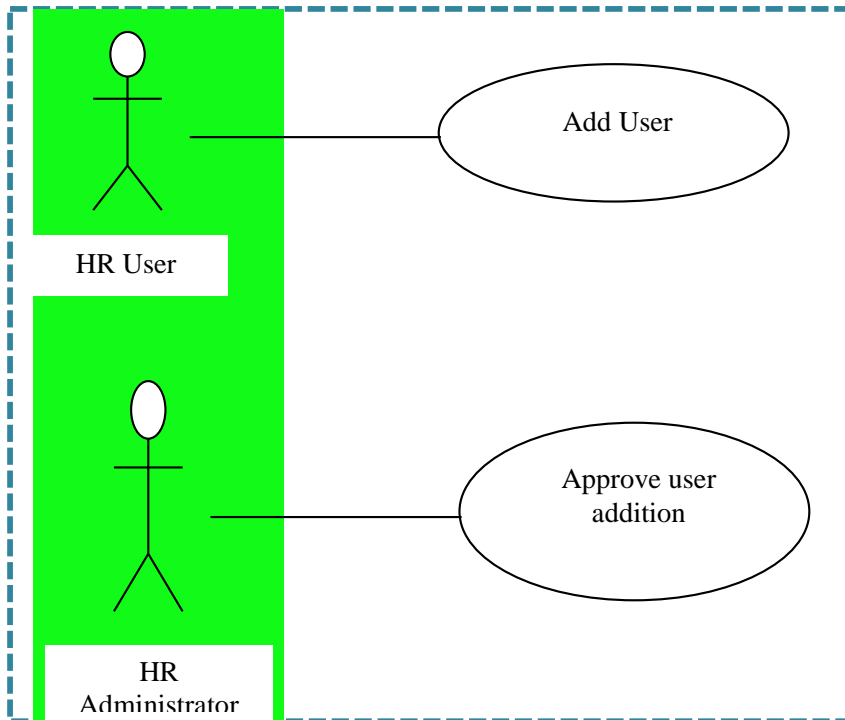
- Feature includes a new screen ("User's Information screen")
- The visualization can be done through the "Users" button from the top menu of the main screen.

#### Initial Step-By-Step Description

User is logged in the system

### 2.2.1.2 Add User

Diagram:



#### Brief Description

Human resource users will be able to add new users to the system, which should appear in the user's view of the system (see use case 1). Users can be added directly on the Users screen or through a widget available on the main application screen.

- Feature includes a new widget on the main screen

#### Initial Step-By-Step Description

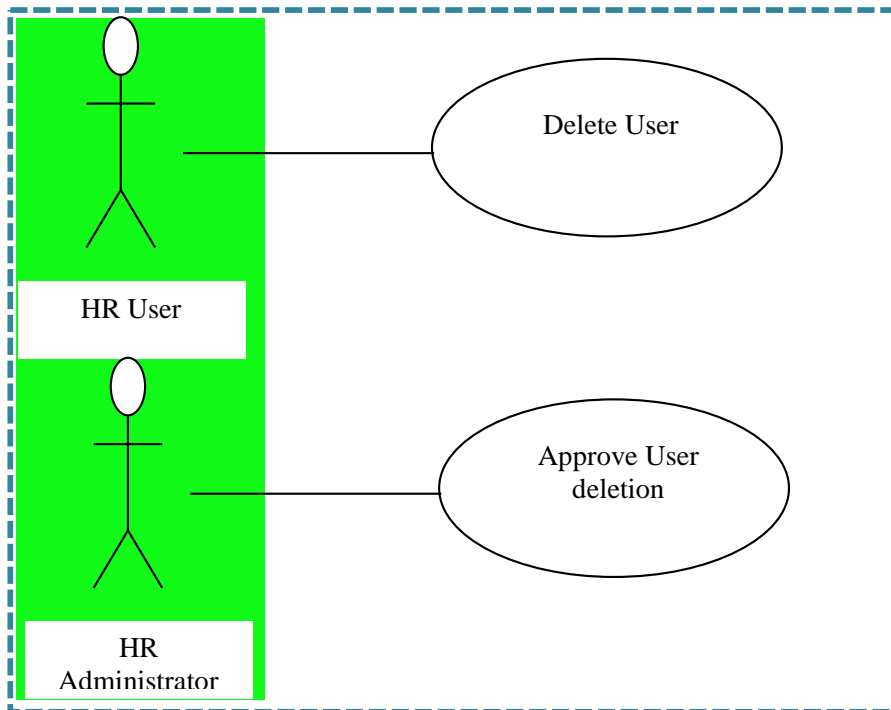
Once connected to the system

1. User (from the human resources sector) navigates to the main screen and clicks the "Add new users" widget available in the footer of the page.  
**Note:** If the user does not have access permission, the button should not be displayed.
2. A screen with a form for creating a new user is displayed.
3. After confirming the data, the new user is temporarily created, being available after HR administrator approval.

Xref: Section 3.2.2

### 2.2.1.3: Delete User

Diagram:



#### Brief Description

A human resources worker may delete an existing user. Such operation will require the approval of the administrator to be completed.

#### Initial Step-By-Step Description

With the user logged into the system

1. A user in the HR views a user
2. When you select the user, the delete option is displayed
3. After clicking delete, an email will be sent to the administrator alerting you to such action
4. The deletion will remain "on hold"
5. The administrator can view the user and decide whether the deletion action will be approved or not.

Xref: Section 3.2.2

### **2.3** *User Characteristics*

Human resource users in the "User" group are expected to access the functionality on a daily basis. Given the number of employees in the company, the number of users will grow as the system becomes more common.

Users of the "Administrator" group need to focus on several tasks per day, so the use of this functionality should not be on a daily basis. E-mail notifications should be sent to these users.

### **2.4** *Non-Functional Requirements*

The new functionality should be implemented using XYZ technology, which is already used in the existing application. The database used will be AAA, which is also used by the current system.

Although the number of accesses to the application is not great, it must be carried out in a way that provides easy understanding with the components that compose it.

### 3.0. Requirements Specification

#### 3.1 External Interface Requirements

N/A

#### 3.2 Functional Requirements

The Logical Structure of the Data is contained in Section 3.3.1.

##### 3.2.1 View User

<b>Use Case Name</b>	View User
<b>XRef</b>	Section 2.2.1.1, View User
<b>Trigger</b>	User in the Users menu and search for a specific user
<b>Precondition</b>	User is logged in User has access to the page User being searched exists
<b>Basic Path</b>	<ol style="list-style-type: none"><li>1. User logs on the system</li><li>2. User navigates to the "Users" page through the main menu</li><li>3. Search for the name of a user</li><li>4. Select the user from the list found</li></ol>
<b>Alternative Paths</b>	For item 2, the user can access the page through a widget on the main screen. Also, a given user can see its own data by clicking in its name in the top right side of the screen.
<b>Postcondition</b>	User details should be displayed on a page in a form.
<b>Exception Paths</b>	The user may not click on one of the search results, not displaying any details
<b>Other</b>	User information includes: <ul style="list-style-type: none"><li>• Name</li><li>• Id</li><li>• Office location</li><li>• Start date in the company</li><li>• End date in company</li><li>• Phone number</li><li>• E-mail</li><li>• Role</li><li>• Project</li><li>• Supervisor</li><li>• Photo</li></ul>

### 3.2.2 Add User

<b>Use Case Name</b>	Add User
<b>XRef</b>	Section 2.2.1.2, Add User
<b>Trigger</b>	The human resources user clicks "add user"
<b>Precondition</b>	The user is in the "Users" screen and has the necessary permissions
<b>Basic Path</b>	<ol style="list-style-type: none"> <li>1. User logs on the system</li> <li>2. User navigates to the "Users" page through the main menu</li> <li>3. Click on "add user"</li> </ol>
<b>Alternative Paths</b>	For item 2, the user can access the page through a widget on the main screen
<b>Postcondition</b>	A new user is created
<b>Exception Paths</b>	<p>The action can be abandoned at any time.</p> <p>Any of the required fields are not filled in or are filled in incorrectly (display error message in red below the required field).</p>
<b>Other</b>	<p>User information includes:</p> <ul style="list-style-type: none"> <li>• Name</li> <li>• Id</li> <li>• Office location</li> <li>• Start date in the company</li> <li>• End date in company</li> <li>• Phone number</li> <li>• E-mail</li> <li>• Role</li> <li>• Project</li> <li>• Supervisor</li> <li>• Photo</li> </ul>

<b>Use Case Name</b>	Approve user addition
<b>XRef</b>	Section 2.2.1.2, Add User
<b>Trigger</b>	The user clicks "add user", an email is sent to the human resources administrator.
<b>Precondition</b>	A user has been added by a HR user.
<b>Basic Path</b>	<ol style="list-style-type: none"> <li>1. Log on to the system</li> <li>2. Navigate to the "Users" page through the main menu</li> <li>3. A previously deleted user appears on your screen with highlight.</li> <li>4. Click the user to view</li> <li>5. Click on approve addition</li> </ol>
<b>Alternative Paths</b>	For item 2, the user can access the page through a widget on the main screen
<b>Postcondition</b>	The user is added
<b>Exception Paths</b>	The action can be abandoned at any time.
<b>Other</b>	N/A

### 3.2.3 Delete User

<b>Use Case Name</b>	Delete User
<b>XRef</b>	Section 2.2.1.3 Delete User
<b>Trigger</b>	The user clicks on "delete user"
<b>Precondition</b>	The user is viewing a user in the "Users" screen and has the necessary permissions.
<b>Basic Path</b>	<ol style="list-style-type: none"> <li>1. User logs on the system</li> <li>2. User navigates to the "Users" page through the main menu</li> <li>3. Search for the name of a user</li> <li>4. A user is selected</li> <li>5. Click on "delete user"</li> </ol>
<b>Alternative Paths</b>	For item 2, the user can access the page through a widget on the main screen
<b>Postcondition</b>	An email is sent to the Human Resources administrator
<b>Exception Paths</b>	The action can be abandoned at any time.
<b>Other</b>	N/A

<b>Use Case Name</b>	Approve user deletion
<b>XRef</b>	Section 2.2.1.3, Delete User
<b>Trigger</b>	The user clicks "delete user", An email is sent to the human resources administrator.
<b>Precondition</b>	A user has been deleted by a user. User belongs to HR Administrator group
<b>Basic Path</b>	<ol style="list-style-type: none"> <li>1. Log on to the system</li> <li>2. Navigate to the "Users" page through the main menu</li> <li>3. A previously deleted user appears on your screen with highlight.</li> <li>4. Click the user to view</li> <li>5. Click on approve deletion</li> </ol>
<b>Alternative Paths</b>	For item 2, the user can access the page through a widget on the main screen
<b>Postcondition</b>	The user is deleted
<b>Exception Paths</b>	The action can be abandoned at any time.
<b>Other</b>	N/A



### 3.3 Detailed Non-Functional Requirements

#### 3.3.1 Data Description

The data descriptions of each of these data entities is as follows:

##### Author Data Entity

Data Item	Type	Description	Comment
Name	Text	Name of principle author	Required
ID	Integer	ID number of Historical Society member	Required Auto-generated Used as key in Database
Office location	Text	Location of office location	Required
Start Date in the Company	Date	Date in which user has joined the company	Required
End Date in the Company	Date	Date in which user has left the company	Can't be less than Start Date
Phone Number	Integer	Employee contact number	
Email Address	Text	Internet address	Required
Role	Text	Employee Role in the company	Required
Project	Text	Project to which user is related	
Supervisor	Text	Supervisor of the employee	
Photo	PNG	Photo of the user	

#### 3.3.2 Security

Only users within the human resources group, whether in the User or Administrator subgroup, will be able to access the new functionality, except by the edit, which can be done by any user, given its own information (not related to HR).



# APPENDIX D – User Story Coding of Document 1

Here are presented the User Story coding for Document 1.

## Create new Finance Diary Screen

### Details

---

Type: Story

Status: CLOSED

Priority: Medium

Fix Version: 2019.1

Label: None

Epic Link: Finance\_Diary

Story Points: 5

Acceptance Criteria:

- Validate that button "Finance Diary" is available in the top menu of the screen.
- When you click on the menu, a new screen called "Finance Diary" is opened.
- Only Finance and Directors have access to this screen.

### Description

---

The Finance Diary view will be displayed through the main screen by clicking the "Finance Diary" button (currently non-existent) located in the top menu of the screen. When you click on the menu, the new screen is opened.

Figure 37: Document 1 - User Story 1 coding

Source: Author

## Create new Calendar in Finance Diary Screen

### Details

Type: Story

Status: CLOSED

Priority: Medium

Fix Version: 2019.1

Label: None

Epic Link: Finance\_Diary

Story Points: 5

#### Acceptance Criteria:

- Validate that calendar is displayed with current month when access the page.
- Existing events are pre loaded in the calendar according to the colors.
- Validate that Past events are gray
- Validate that Holiday type events are green
- Validate that Payment events are Blue
- When clicking in an event, validate the information in the pop-up

### Description

The new Finance Diary screen should be displayed containing a calendar with the current month and all events loaded in it.

The user can navigate through the calendar months by clicking on the arrows.

When clicking on an event, a Pop-Up should appear showing the event information.

Calendar events should be colored according to the following rules:

- Gray: Events in the past
- Green: Holiday Events
- Blue: Events of type Payment

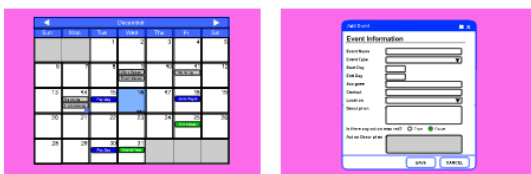


Figure 38: Document 1 - User Story 2 coding

Source: Author



# APPENDIX E – User Story Coding of Document 2

Here are presented the User Story coding for Document 2.

## Investment

### Details

---

Type: EPIC

Status: CLOSED

Priority: Medium

Label: None

### User Stories

TICKET-111 - As a user I want to see all my investments in one screen

TICKET-112 - As a user I want to change my investments priority

Mentioned in: [Investment Screen](#)

### Description

---

This User Story refers to the new "My Investments" screen, which is intended to enable the user to manage their financial investments. In this new screen the user can view all his current investments, grouping them by name of the investment showing both the name and the value currently invested, as well as the amount of profit (being highlighted in green) or loss (being highlighted in red). The screen should have options for viewing details of the investment as well as options for filtering the investments by name or by invested amount.

### Attachments

---



Document.pdf

Figure 39: Document 2 - Epic coding

Source: Author



## As a user I want to see all my investments in one screen

### Details

Type: Story

Status: CLOSED

Priority: Medium

Fix Version: 2019.1

Label: None

Epic Link: Investment

Story Points: 5

#### Acceptance Criteria:

1. Ensure that the screen is accessible through the menu "My Investments"
2. Ensure that all investments are listed, initially in alphabetical order
3. It should be possible to filter the investments and sort them by name or value
4. Profits should be shown in green color
5. Prejudices should be shown in red color
6. Field to display application information should be available
7. Validate investment information (details)

#### Consists of:

TICKET-111 - Create UI

TICKET-112 - Database development

Mentioned in: Investment Screen

### Description

Creation of a new screen, through which the user can view all its current investments, grouping them by name of the investment showing both the name and the value currently invested, as well as the amount of profit (being highlighted in green) or loss (being highlighted in red).

Add view details and filtering options for the investments.

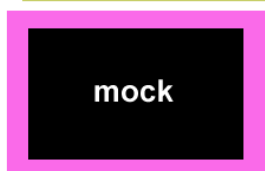


Figure 40: Document 2 - User Story 1 coding

Source: Author

**MENU**

- Project
  - Fontend
  - Security
  - Project overview
  - Investment screen
  - Team contacts

### Investment Screen

This User Story refers to the new "My Investments" screen which is intended to enable the user to manage their financial investments. In this new screen the user can view all its current investments, grouping them by name of the investment showing both the name and the value currently invested, as well as the amount of profit (being highlighted in green) or loss (being highlighted in red). The screen should have options for viewing details of the investment as well as options for filtering the investments by name or by invested amount.

Document.pdf

Tickets

TICKET-111 - As a user I want to see all my investments in one screen

TICKET-112 - As a user I want to change my investments priority

Figure 41: Document 2 - Confluence 1 coding

Source: Author

**MENU**

- Project
  - Fontend
    - Documentation
  - Security
  - Project overview
  - Team contacts

### FRONTEND DOCUMENTATION

Icon	Font	Name	Description										
	Original	Btn_filter1	Button shown when a column is filtered. Btn_filter2's derivation	<b>Profit</b> <span style="font-size: 1.2em;">▾</span> <span style="color: green; font-weight: bold;">R\$ 1.500,00</span> <span style="color: green; font-weight: bold;">R\$ 300,00</span> <span style="color: red; font-weight: bold;">R\$ 100,00</span>									
	Original	Btn_filter2	Button shown when a column can be, but is not, filtered Btn_filter1's derivation										
	Bootstrap 2.3	Btn_plus	Button shown when a line can be expanded to show more information. Btn_minus's derivation	<div style="font-size: 0.8em; border: 1px solid gray; padding: 2px;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid gray;"> <span>Name of investment ▾</span> <span>Invested value ▾</span> <span>Profit ▾</span> </div> <div style="padding: 2px;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid gray;"> <span>Company A</span> <span>RS 10.000,00</span> <span>RS 1.500,00</span> </div> </div> </div>									
	Bootstrap 2.3	Btn_minus	Button shown when a line can be retracted to show less information. Btn_plus Derivative	<div style="font-size: 0.8em; border: 1px solid gray; padding: 2px;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid gray;"> <span>Company C</span> <span>RS 2.000,00</span> <span>RS 100,00</span> </div> <div style="padding: 2px;"> <table style="width: 100%; border-collapse: collapse; font-size: 0.7em;"> <tr> <td style="width: 50%;">Investment</td> <td style="width: 25%;">Apr /18/ RS 1.000,00</td> <td style="width: 25%;">RS 0</td> </tr> <tr> <td></td> <td>May /18/ RS 500,00</td> <td>RS 150,00</td> </tr> <tr> <td></td> <td>Jun /18/ RS 500,00</td> <td>RS 250</td> </tr> </table> </div> </div>	Investment	Apr /18/ RS 1.000,00	RS 0		May /18/ RS 500,00	RS 150,00		Jun /18/ RS 500,00	RS 250
Investment	Apr /18/ RS 1.000,00	RS 0											
	May /18/ RS 500,00	RS 150,00											
	Jun /18/ RS 500,00	RS 250											
	Bootstrap 2.3	Btn_Home	Button symbolizing the home page of the application.	<div style="font-size: 0.8em; border: 1px solid gray; padding: 2px;"> <div style="display: flex; justify-content: space-between; border-bottom: 1px solid gray;"> <span>MY ACCOUNT</span> <span>ACTIONS</span> <span>MY INVESTMENTS</span> </div> <div style="display: flex; justify-content: space-between; align-items: center;"> <span>HOME</span> <span>SEARCH</span> <span>USER_123</span> </div> </div>									

Figure 42: Document 2 - Confluence 2 coding

Source: Author

# APPENDIX F – User Story Coding of Document 3

Here are presented the User Story coding for Document 3.

## Create User Screen

### Details

Type: Story

Status: CLOSED

Priority: Medium

Fix Version: 2019.1

Label: None

Epic Link: User

Story Points: 5

#### Acceptance Criteria:

1. Display all valid records
2. Display name, id, office location, start/end date in company, phone number, e-mail, role, project, supervisor and photo.
3. Validate search
4. When clicking back button after search, the previous search should be loaded.
5. Validate edit options for user and HR users. Invalid fields should show message in red below the field.

#### Consists of:

- TICKET-111 - Create New screen with edit operation
- TICKET-112 - Create database side
- TICKET-113 - Create search option

### Description

As an user I want to be able to search a user in the system, so that I can see the information of that user.

As an user, I want to see my own information by clicking in my name in the top of the screen.

In the top menu, create a new item (Users). The new screen should have a search from which the users can be found. When finding a user and clicking in it, the user details should be displayed.

User personal information can be edited the user itself, other information are only editable by the RH users.

### Commentaries

From: John Titor

See mock [IN THIS LINK](#)

Figure 43: Document 3 - User Story 1 coding

Source: Author

## Implement 'add user' functionality

### Details

Type: Story

Status: CLOSED

Priority: Medium

Fix Version: 2019.1

Label: None

Epic Link: User

Story Points: 5

Acceptance Criteria:

1. Display all valid records
2. Display name, id, office location, start/end date in company, phone number, e-mail, role, project, supervisor and photo.
3. Validate mail is sent to HR admin
4. HR admin should be able to approve add user operation
5. Validate that user is not visible to non HR users before the same has been approved by HR admin user.

### Description

As a HR user I want to be able to add a new user to the system.

As an HR admin user, I want to receive 'an email when a new user is added in the system, so that I can be aware that I need to approve it.

As an HR admin user I want to be able to approve new users added in the system so that I can verify all the information before it is available to everyone.

Given the User screen, HR users can add a new user in the system. After confirming user information, an email is sent to HR Admin so the admin can approve the new user being added to the system.

### Commentaries

From: John Titor

See mock [IN THIS LINK](#)

Figure 44: Document 3 - User Story 2 coding

Source: Author



# APPENDIX G – Artifacts Dispersion

The structures here present illustrate the way the information is stored and handled inside the projects. The main purpose of this mapping is to know where the documents, earlier studied, were being used. Beneath this purpose, sits the interest in discovering which information, may it be written or an artifact, is used somewhere else, and how it is dispersed among the different ways of communication presented in the projects (Jira<sup>1</sup>, Confluence<sup>2</sup>, etc.).

This document presents an overview with some notes that complement the findings presented in the code mapping (see [A](#) to [F](#)).

## G.1 Artifacts Dispersion - Document 1

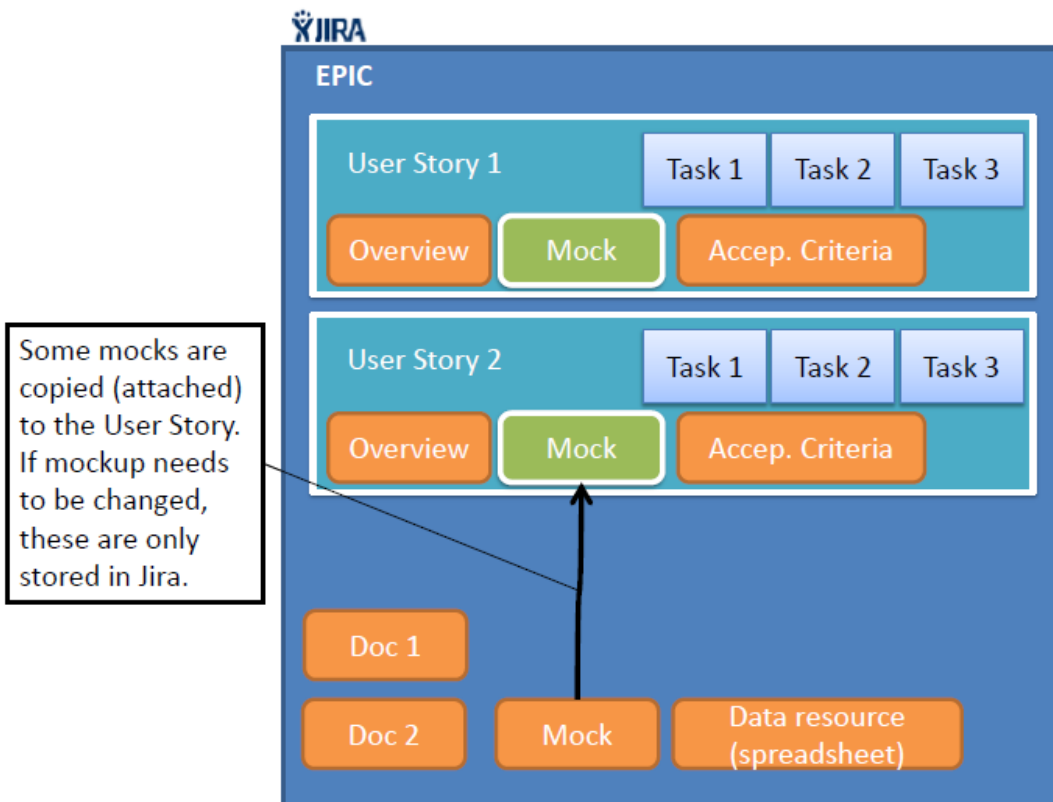
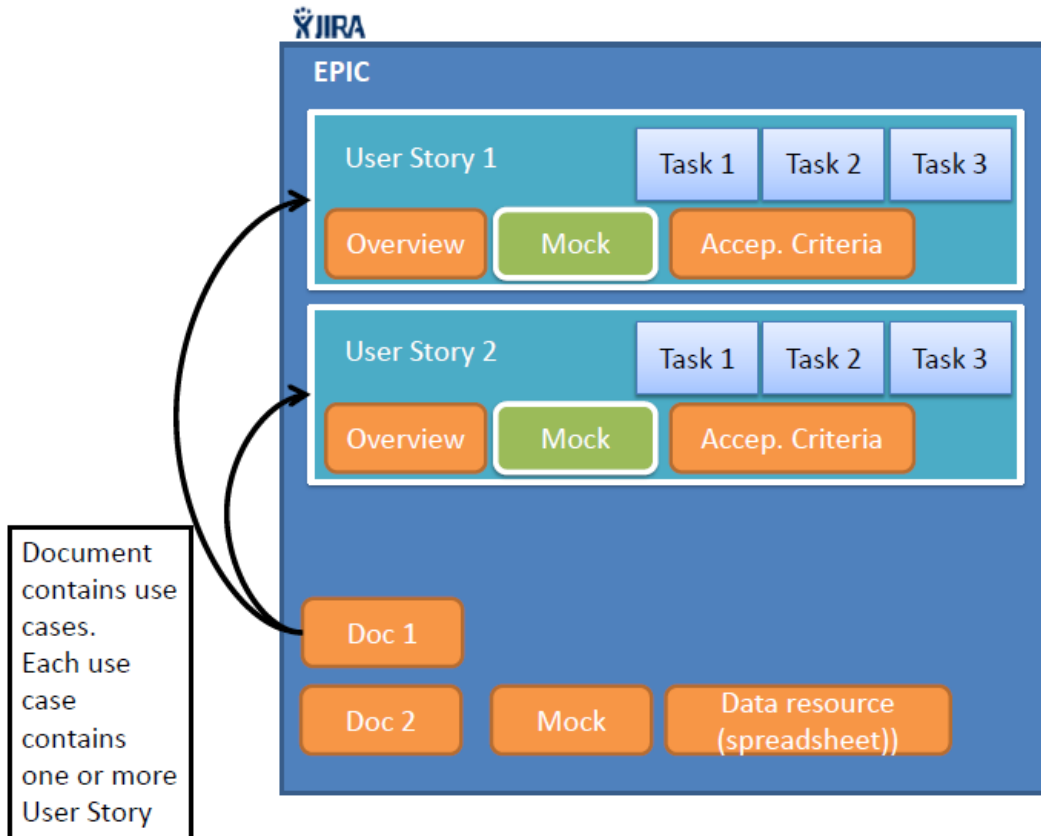
Artifacts Dispersion for document 1.

---

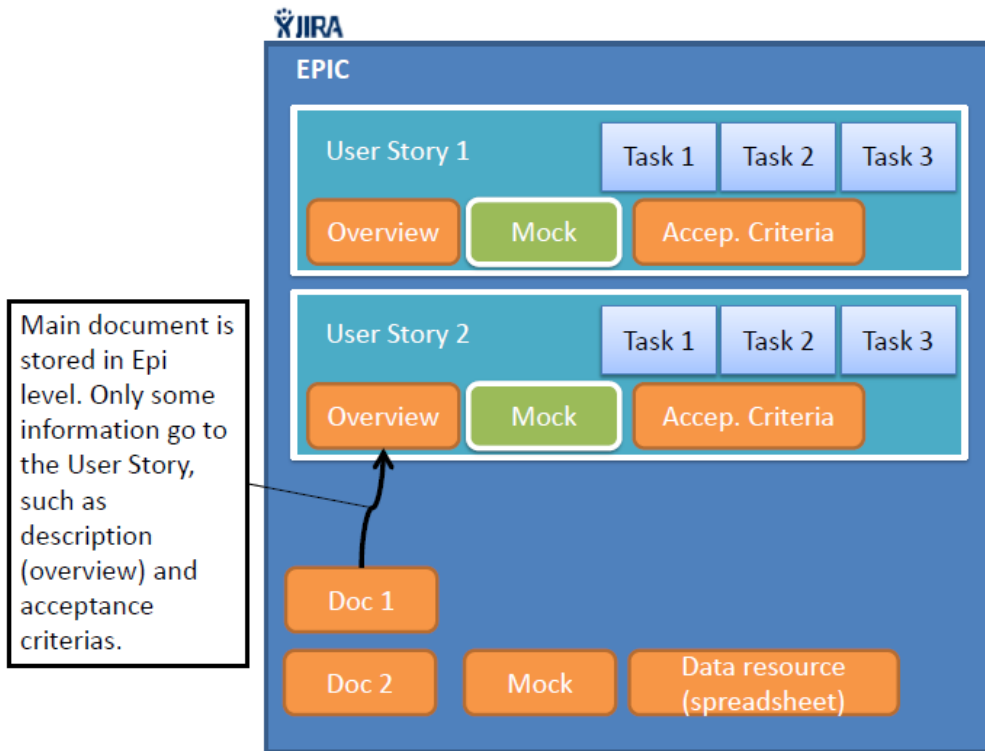
<sup>1</sup> JIRA: <https://www.atlassian.com/software/jira>

<sup>2</sup> Confluence: <https://confluence.atlassian.com>

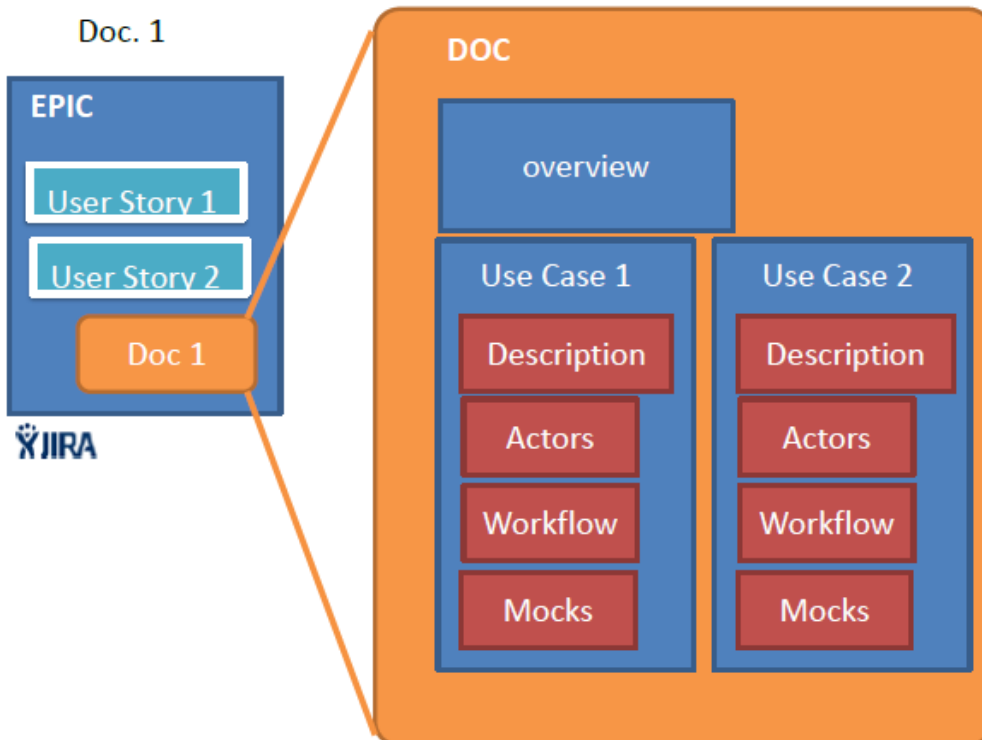
Jira structure overview.







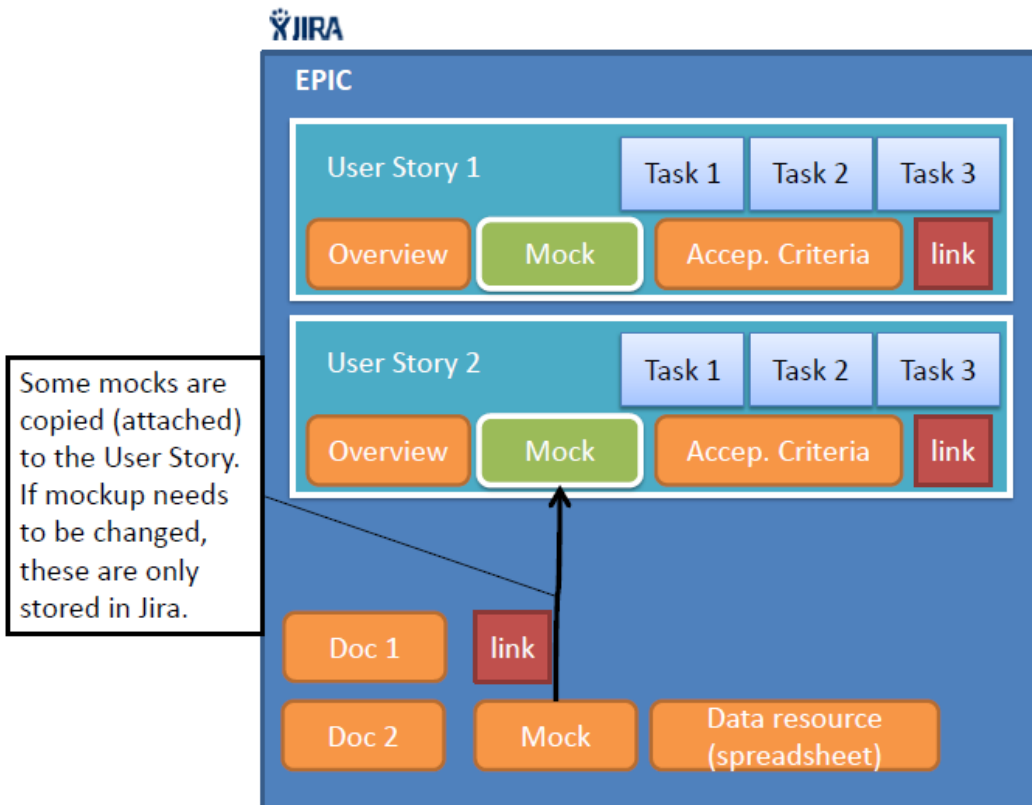
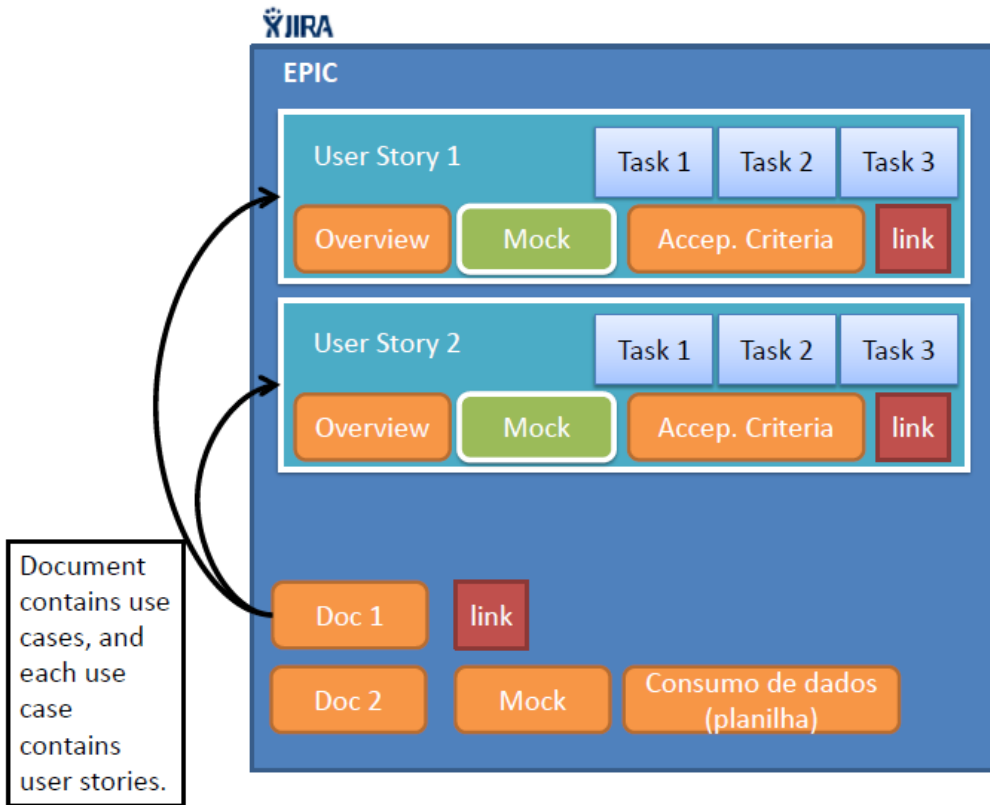
Document and Jira overview. Notice that Document is attached at Epic level.

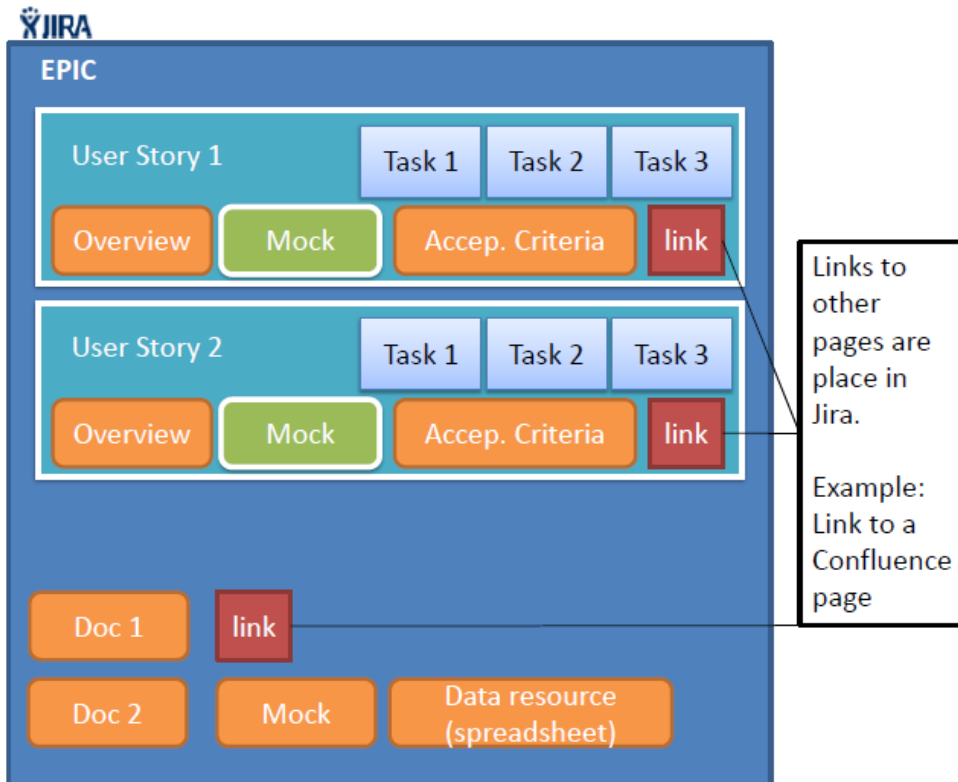


## G.2 Artifacts Dispersion - Document 2

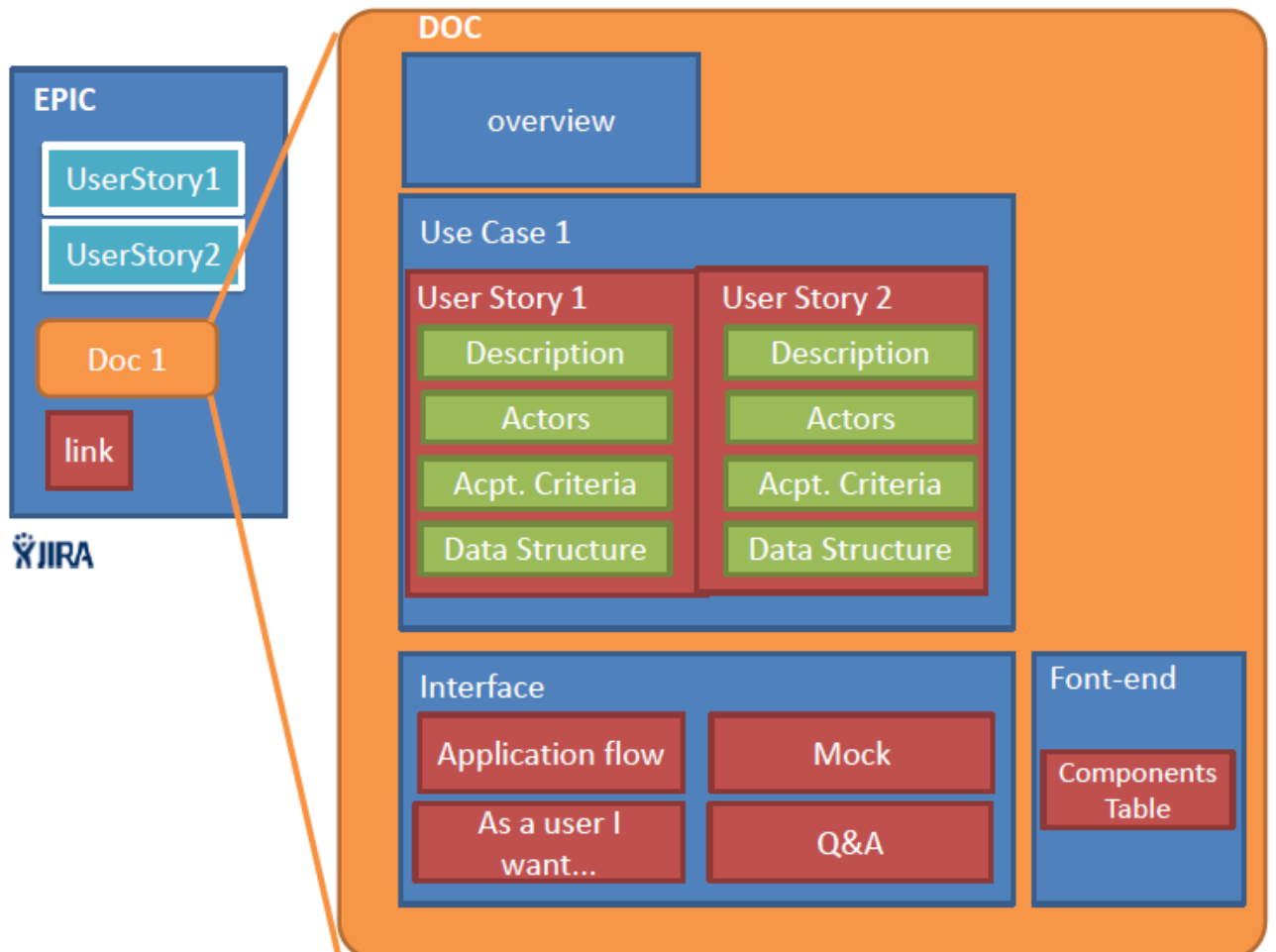
Artifacts Dispersion for document 2.

Jira structure overview.



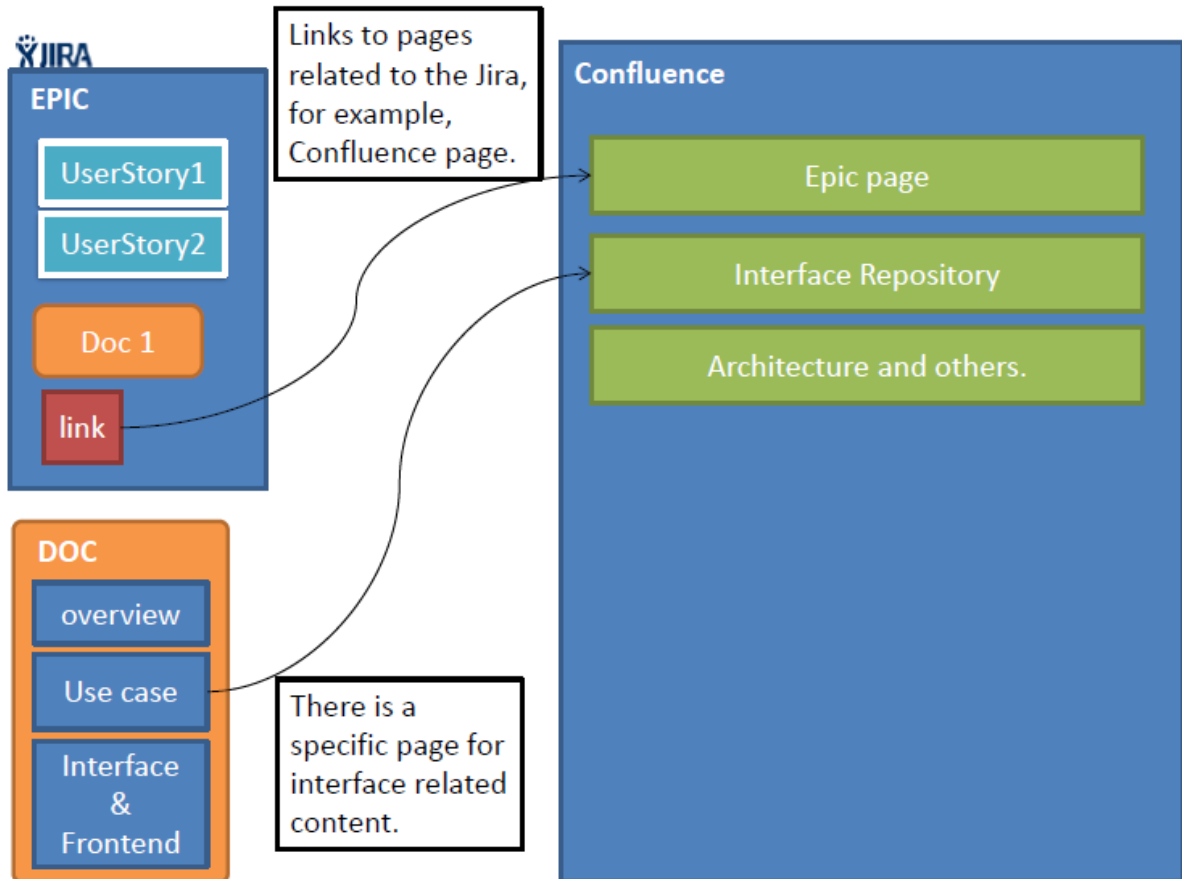


Document and Jira overview. Notice that document is attached to Epic level.



Besides document and Jira, another platform, Confluence, is used by the team. In Confluence, pages regarding software are presented, and information that are not contained in Jira, can be found in there, if not in the document itself. Some information obtained after document creation are only stored in Jira and Confluence.

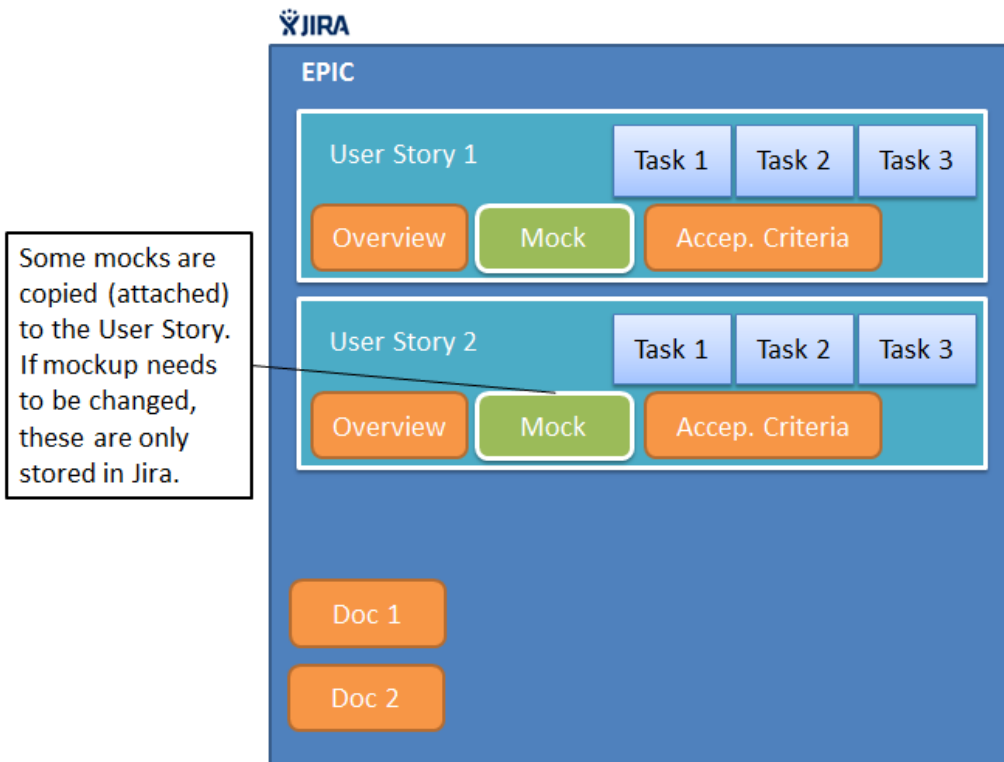
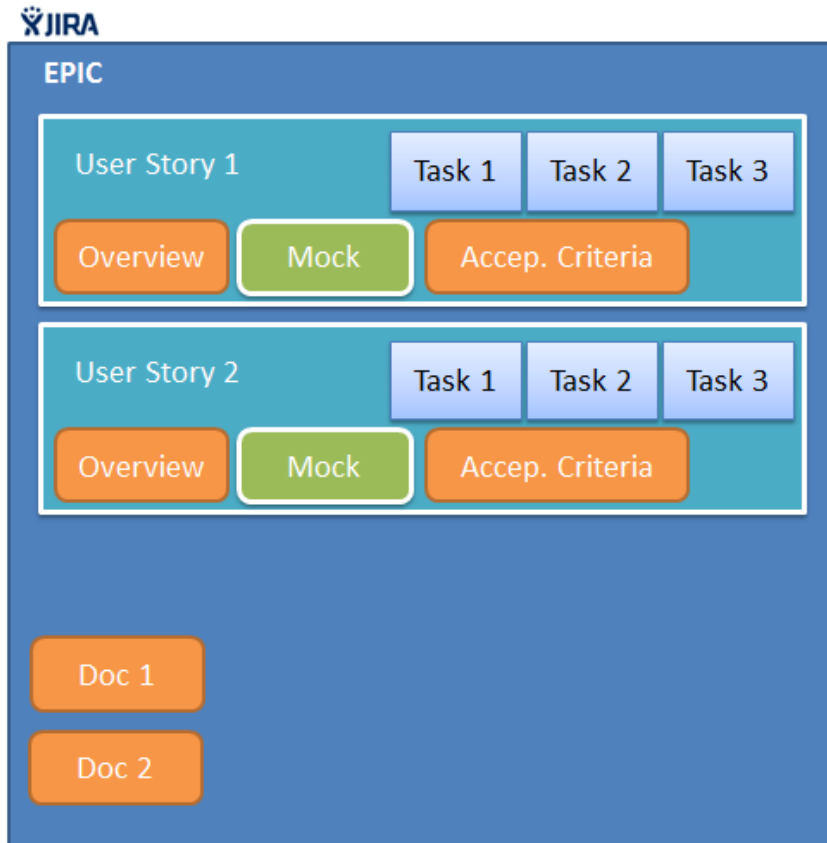
Links to confluence can be found in Jira, at Epic level as well as in user Story level.



### G.3 Artifacts Dispersion - Document 3

Artifacts Dispersion for document 3.

Jira structure overview.



Document is stored at Epic level.

