Navigational Distances between UX Information and User Stories in Agile Virtual Environments

Abner Cleto Filho[a] and Luciana A. M. Zaina[b]
Federal University of São Carlos, Sorocaba, São Paulo, Brazil

Keywords: Agile Practice, User eXperience, User Story, Navigational Distance.

Abstract: User stories (US) are valuable artifacts to agile teams, being a succinct requirement description 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. Many approaches on merging agile and UX are presented in the literature, but few have analysed how UX information and US are related in agile virtual environments. This paper presents a case study which investigates the navigational distances between UX information and USs. The goal was to understand how agile practitioners linked UX information to USs. To do this, we conducted a qualitative analysis in 13 requirement documents of three different industry projects and also we explored the USs derived from such documents. We propose a classification of navigational distances found among UX information and USs, discussing the pros and cons of each distance type. Our work contributes to motivate agile teams in rethinking about different forms to organize the information, artifacts and documents in virtual environments.

1 INTRODUCTION

Studies on agile and User eXperience (UX) practices are not new in the literature (Brhel et al., 2015; Da Silva et al., 2011; Schön et al., 2017). Issues about the integration of these two areas have changed through time. A recent study by Da Silva et al. (2018) concluded that agile and UX are getting closer over time, and the tendency is that both merge into a single practice. The same study still reinforces the differences among UX and agile, reporting context-dependency issues, where different teams use a diversity of artifacts and techniques to create shared understanding.

UX and agile practices are often seen from different lenses. Agile aims to have small pieces of work delivered to clients in a fast paced manner, receiving feedback constantly (Brhel et al., 2015; Trienekens et al., 2018). On the other hand, UX aims to work with the users emotions and perceptions on the product it is using (Garrett, 2010), being it main focus to enrich the quality of the product, working with non-functional aspects and to create a holistic view about the product usage (Hassenzahl and Tractinsky, 2006). These different ways of working introduce challenges regarding on how agile teams manage the relationship of agile and UX documentations.

A diversity of virtual and physical tools have arisen to support team-work. Virtual environments (e.g. Jira[1]) are widely employed for managing the diversity and the great amount of information that is created by agile teams (Akman et al., 2015). User Story (US) is recognized as the most valuable artifact to agile teams (Cohn, 2004). Its value comes from it gives details about software features as well as supports individuals in managing the team tasks (Schön et al., 2017). Although commonly used, it is known that USs may not contain all the information an agile team needs for the software development (Hess et al., 2017; Kashfi et al., 2017). Agile teams often use other documents and artifacts to complement the information found in USs (Garcia et al., 2019).

Moreover, agile practitioners have struggled on how to organize the great amount of artifacts that support their work (Hess et al., 2017). The lack of linking among relevant artifacts and information, as well as the excessive number of steps to reach the information, can introduce problems regarding the distance between the artifacts and information.

The literature has reported findings about agile documentation issues and agile-UX integration.

[a] https://orcid.org/0000-0002-7804-0171
[b] https://orcid.org/0000-0002-1736-544X

[1] https://www.atlassian.com/software/jira

Filho, A. and Zaina, L.
Navigational Distances between UX Information and User Stories in Agile Virtual Environments.
DOI: 10.5220/0009354801850192
In *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 2*, pages 185-192
ISBN: 978-989-758-423-7

However, little is known on how UX information are linked to USs in virtual environments, which are often used by agile practitioners. With the aim of investigating this issue, we defined the following research questions (RQs): *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?*.

To answer the RQs, we conducted a case study in a software development company based in Brazil. In our study, first, 13 requirement documents of 3 different projects were analyzed to uncover how the UX information was connected to USs. After, we examined the USs produced from such documents and its relationship with other artifacts. We performed all the investigation considering that USs, requirement documents and artifacts related to them were stored in virtual environments.

Our findings revealed that UX information was spread in different artifacts and in a diversity of ways. Consequently, this dispersion could introduce a problem of navigational distance. As contribution our work presents a categorization of different forms of navigational distance that can be found between US and other artifacts.Our results may support other agile teams in the understanding on how the UX information often is distributed around the virtual environments and how to better organize it. We also provide some recommendation of how agile teams can avoid facing issues caused by navigational distances.

2 FUNDAMENTALS AND RELATED WORK

2.1 Fundamentals

Garrett's framework (Garrett, 2010), named *The elements of UX* (Figure 1), helped us to define what kind of UX information we were looking for in the documents and USs. It is a framework containing five planes describing elements of UX. The planes can be summarized as following.

The strategy plane aligns users needs with the product's objective. The scope plane, defines the functional specification of the product. Next, the structure plane, deals with how the system behaves in a user interaction (i.e. interaction design) and the arrangement of content elements (i.e information architecture). In a further refinement of the structure, the skeleton plane deals with concrete elements such as buttons, fields, menus (interface design), content representation (information design) and with the interaction over user interface (navigation design). Finally, in the surface plane, the aesthetics elements are considered to produce a pleasing interface and fulfills the goals of the other planes. In each plane, Garrett suggests some UX artifacts that could be used to support developers in working on each element of UX.

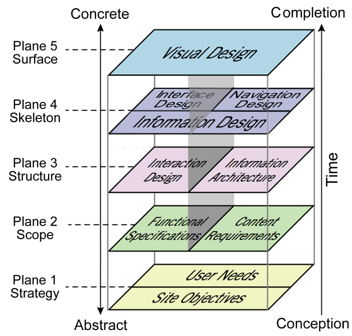

Figure 1: The Elements of UX - Adapted from (Garrett, 2010).

To explore the requirement documents and USs we adopted the coding technique, a qualitative technique that relates codes to chunks of text. These codes receive denominations that give meaning to the chunks of texts they refer to (Bohm, 2004). Later, these codifications are revisited and grouped when patterns of information are identified. For this analysis, we performed three rounds of coding using *closed coding* and *open coding* approaches.

In the *closed coding* the researcher searches for chunks of text that match with some concepts that the researcher previously have. In our case, we used Garrett's Framework as our concepts (Garrett, 2010). We also adopted the *open coding* approach. In this case, no pre-conception about data labelling is used. The researcher created new codes that bring meaning to the chunk of text. Considering the exploration of the text, first we examined each word or line and coded it. After, larger pieces or blocks of text are explored, and new codes emerged from this analysis (Bohm, 2004).

As our main goal was to investigate the linking between US and UX information, our study was based on the theory of distances (Bjarnason and Sharp, 2017), which aims to categorize the difference of position or level among the stakeholders or artifacts 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 artifacts are regarding their positions (Bjarnason and

Sharp, 2017). In a virtual tool, as Jira, such distance can be taken as the length of the path to navigate between the artifacts, being the length calculated by the number of links that the individuals have to access to reach the information (Bjarnason and Sharp, 2017).

2.2 Related Work

Kashfi et al. investigated the challenges software practitioners face on dealing with UX in software development (Kashfi et al., 2017). Among the different challenges, the authors identified the need for mechanisms that improve the traceability between UX-related and other requirements. Schon et al. carried out a survey and presented the key challenges in agile requirements engineering. One key challenge reported was how to manage the diversity of documentation that support agile teams in their work (Schön et al., 2017). Liskin 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).

Kassab, from a survey that compares agile with waterfall requirements engineering, points that Jira is the most used tool to support agile practices (Kassab, 2014). Hess and Seyff report that agile approaches have a strong focus on face-to-face communication instead of having detailed Requirement Engineering (RE) documentation (Hess et al., 2017). This emphasizes its early analysis, which showed that artifacts used in agile activities cover key requirements information, but its has a lower detail level if compared with 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), many works have proposed ways of how to write USs. In 2009, Cohn proposed a grammar as a pattern to the writing of user story 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 extended Cohn's proposal, creating the UserX story, from which agile practitioners can add UX information in the US body (Choma et al., 2016). The authors concluded that agile practitioners have difficulties in understanding how a single US can hold all the UX information they need.

In a literature mapping about agile and UX practices, Garcia et al. concluded that virtual artifacts are mostly used as basis for development phases in agile practices (Garcia et al., 2017). The authors also reported that mock-ups and USs are often used in combination to support the agile developers work. Such conclusion is also supported by Garcia et al. (2019), which carried an investigation in an online agile community, also concluding that mock-ups are adopted in combination with USs (Garcia et al., 2019). However, the studies above do not cover how the UX information outside the US can still be linked to it.

3 CASE STUDY SETTINGS

The case study was conducted in a software global organization of financial domain. It is present in twelve countries, having more then 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 for 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 this organization, USs are written based on software requirement documents. The process of requirement specification until the writing of USs often follows the same steps. Briefly, the US writing process can be described as following: First, the requirements are raised by the interaction of the Project Owner (PO) with stakeholders (end-users, managers, etc.). Such requirements are formally written in the requirement documents format, which is later sent to development team. Later, 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 questions that the development team can have about the requirement 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. which is a large US that cannot be delivered within a single iteration (Cohn, 2004). The Epics are stored in Jira virtual tool, which is a commonly used in agile practices (Kassab, 2014). Frequently, the USs are linked to other complementary documents that give information about UX or functional requirements. These complementary information are stored in different virtual repositories. UX information is found in extra documents, stored into Confluence[2] or in SharePoint[3].

In our investigation, we analyzed 13 requirement documents from 3 different projects related to the financial domain, mainly focused on serving as reports for financial control and risk mitigation. Each project have about 10 people involved, being these developers, project leaders and quality assurance specialists. None of the projects had a UX expert with exclusive dedication to the projects.

Despite the present study has been done over virtual tools used by the teams (Jira, Confluence and SharePoint), 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 USs and other documents were stored in a virtual environment and not exist as tangible documents (i.e. physical documents). Also, given data confidentiality issues, no real data will be presented in this study. To illustrate how we conducted our analysis, we will take fictitious data, presented as documents and artifacts. For each of the three studied projects, one sample document was created, along with US and its relations to external tools. Three examples of documents in PDF format are available in the link here[4].

4 ANALYSIS APPROACH

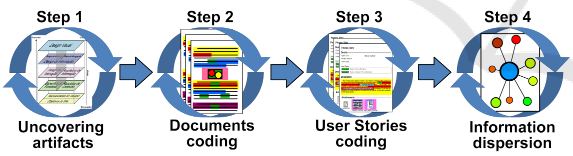

Figure 2: Steps of the Analysis.

Considering the underpinnings presented in Section 2.1, a qualitative analysis was carried out by two experts in UX and Software Engineering: (i) a master postgraduate student that has about five years of experience in agile practices in the industry; and (ii) a senior researcher with over 20 years of experience in UX and software practices. Figure 2 illustrated the four steps we followed in the analysis. After the master postgraduate student performed a step, the senior researcher revised and refined the results.

In Step 1 - Uncovering Artifacts, we did not performed any coding process. 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[5]. After exploring the documents sections that did not contain UX information (e.g. database structures), were removed from the documents. Given only UX information were relevant for our study purpose, and to avoid misunderstandings during the analysis, we decided to remove these not useful sections from the documents. Next 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.

A total of 13 requirement documents, containing a total of 68 use cases were analysed. From these, 45 use cases contained aspects related to UX elements.

The process of coding the documents (step 2) was conducted in the 13 documents selected in the step 1. The coding process was performed in three steps. First, we applied the *closed coding* approach supported by Garrett's framework (Garrett, 2010). We labelled a chunk of a document when we found out an evidence of UX information. The codes assigned reflected the elements of UX of Garrett's Framework. 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. Finally, we performed a double check in all the documents to search for inconsistencies. Figure 3 shows an example of labelling process.

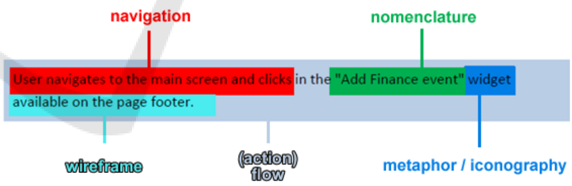

Figure 3: Example of Coding Process.

In Step 3 - User Stories coding, we examined the USs following the same proceedings performed in step 2. Given that USs were written based in the requirement documents, we considered that we could find the same UX information found in the documents embedded in the USs or related to them. In this company the USs were stored only in virtual tool (i.e. Jira). We carried out the coding process having the USs as our starting point, also examining whether that US had connections to other virtual tools. Figure 4 shows an

[2]Confluence: www.atlassian.com/software/confluence

[3]SharePoint: https://products.office.com/sharepoint

[4]drive.google.com/drive/folders/1o4251cGs6eftjSGP0D99yawqbZk4O2gM

[5]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).

example of coding in a US.

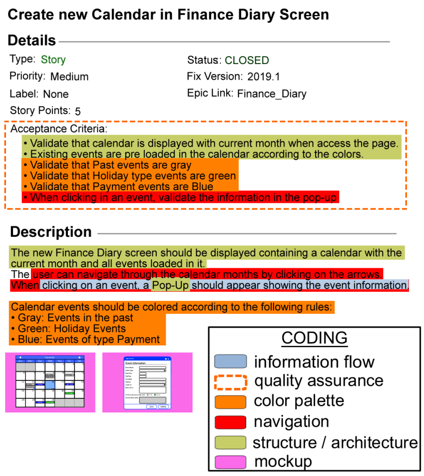

Figure 4: Example of Coding Process Did in a User Story.

Step 4 considered the step 3 outcomes to explore the UX information dispersion in the virtual environments. We noticed that UX information was spread in different virtual tools. To examine how the UX information was hold into the USs, we explored in which way the information was connected to them. Such connections could be found through hyperlinks, attachment, mentions, and so on.

5 FINDINGS

As a final result of our 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.

5.1 Connections between US and UX Information

In Figures 5, 6 and 7, we can see where the UX information relates to USs in the different projects we explored in this study. The connections between two elements represent the ways in which the 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).

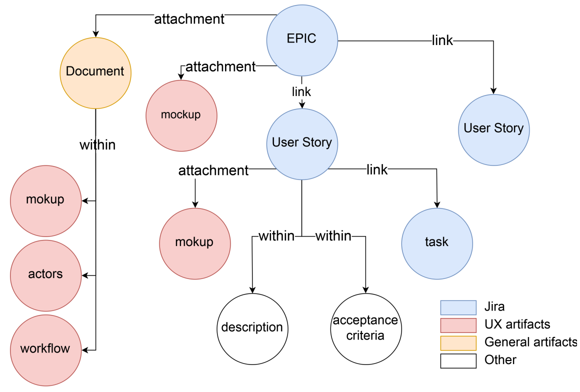

Figure 5: Software Project 1 - Information Distribution.

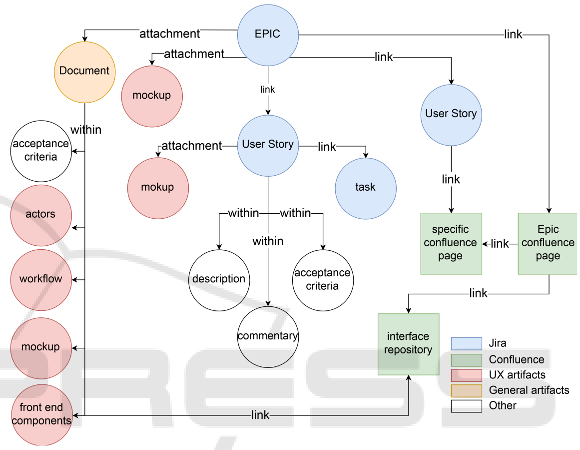

Figure 6: Software Project 2 - Information Distribution.

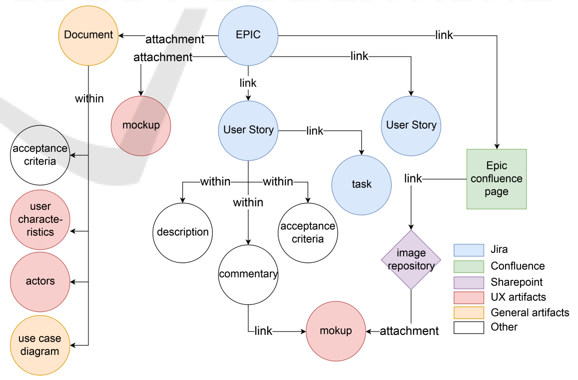

Figure 7: Software Project 3 - Information Distribution.

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 an 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, 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 (Liskin, 2015). Poor link management could lead to the artifacts not having their relationship properly established, causing a disruption in further information retrieval processes.

Finally, *attachment* represents when an artifact is attached to another. This type of connection is commonly found in container artifact (Liskin, 2015). Attachments can be directly accessed, which makes the attached artifact to be an easy way to obtain detailed information. However, the attached elements can only exists 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.

5.2 Navigational Distances Types

Considering the diagrams presented in Figures 5 to 7, we applied the principles of graph theory (Harris et al., 2008) in order to retrieve the distance among the information previously mapped. Instead of using the classification of navigation distances based on number of clicks (Bjarnason and Sharp, 2017), this study proposes a classification for the type of element relationship that the navigation will be performed from. The 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 et al., 2008).

Our results revealed that, for the projects studied, there are not large distances found among the artifacts and UX information. Results showed that the total number of navigation steps required to go to the farthest artifacts containing UX information that relate to each other, varies from 3 to 6 steps. This is an indication that there are no great efforts in order to find information in terms of navigational steps.

In Figures 5 to 7, we notice that to reach information held in UX artifacts from US, developers should have intermediate artifacts or documents they have access first. This result shows that UX information can be hidden when considering USs as the start point for accessing it, given information can be stored outside the US or even within it but not visible at a first glance. The navigational distance to navigate to other tools or open an attachment is present and makes the UX information to require at least a minimal effort to be seen.

From such results, we could classify the navigational distance into four types (see Table 1).

Table 1: Navigational Distances Types.

Type	Description	Characteristics
Zero	UX information is stored in 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 lowest levels in artifact hierarchy (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)

Zero and Infinite distances are extreme cases. In *zero distance*, there is no need to navigate through the artifacts, once all required information is stored within an artifact. The *infinite distance*, named after a graph principle that represents the distance among two not linked points, stands for a navigational distance that is infinite, meaning that there is no relationship among the US and the UX information. In such scenario, the navigation can only happen through a previous knowledge from the developer. With no explicit relationship available, the virtual environment lacks on providing traceability of UX information to the developers. Along with traceability issues, the difficulty on history retrieval may also be present.

For the *deep distance*, UX information is stored in a very detailed level, requiring many navigational steps to reach the information. In such arrangement type, the information granularity can be taken as a pro point. However, whether upper UX information hierarchy levels have good overview, the need to go down to the deeper levels is low. Storing 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 (in a graph-based hierarchy, see Figure 7).

In *wide distance* classification, the UX information is stored in different places within the same hierarchy level, making distances to be spread in width instead of depth. A wide distance type can indicate multiples tools usage or many information being stored in the same hierarchical level of a given tool, which may indicate a lack of information granularity.

Different from *deep distance* type, where detailed information are stored in a deeper level inside the same tool, in *wide distance* the information is placed in a separate location, specific to the type of UX information it relates to (i.e. a repository).

6 DISCUSSION AND RECOMMENDATIONS

By answering our RQs, we can discuss the results we found out. Regarding *RQ1 - How are UX information and USs connected into software virtual environments?*. Our findings revealed that UX information could be found embedded into USs, as well as connected by links or attachments. By having the US holding all the UX information, the developers can get a big picture of UX (Hassenzahl and Tractinsky, 2006). On another hand, USs are made to be kept small, so the pieces of work can be done and delivered in a small period of time (Cohn, 2004). Given requirements need to be broken into small pieces to fit the US structure, the UX information dispersion in many USs is a common scenario in agile practices.

RQ2 - What are the navigational distances found to access UX information from USs into software virtual environments?. From our results, we noticed that different navigational distances can be introduced from USs to UX information. By having the UX information spread in different tools, developers can struggle in having the traceability of such information, which is reported in literature as one of the main problems of UX integration in software development process (Kashfi et al., 2017).

The distance among the UX elements and the other information found in the mapping performed in this study indicates loss of UX big-picture, as it is harder to see all information, or its connections, when it is dispersed (Da Silva et al., 2018). The analysis also showed that UX information was distributed in several tools and artifacts.Considering our findings, we recommend some good practices to reduce the navigational distance between USs and UX information.

To *avoid the UX information duplication*, the differences of UX and agile should be taken into account (Brhel et al., 2015). UX work should be done before the actual development, but not too far from it (Da Silva et al., 2018), and involving developers (Brhel et al., 2015). Given a hierarchy level of storage places, placing "generic" information or artifacts in a higher hierarchy level (i.e. Epic) is a good practice when the same can be shared among several USs. Moreover, whenever this information needs to be changed, the modification can be applied in only one location, diminishing the issue on having different version or outdated information or artifact spread across the virtual environment.

Practitioners should keep the *US linked to UX information* by connecting the different artifacts across the virtual environments. This recommendation intends to avoid the loss of big-picture given the spread information can still be tracked. Spreading UX elements across several environments is fine, as long as it is not duplicated. Overloading USs with much information may cause it to hold information that could be used elsewhere. Therefore we believe that storing UX elements outside USs should be considered an acceptable approach as long as links are created.

The creation of *central repositories* to hold UX artifacts prevents the UX information to be stored in several USs, making the navigational distance to these information to have a limited maximum distance (i.e. the central repository).Central repositories can also be seen as "landmarks" in the virtual environment navigation, making it easier to remember and find an initial step for a more detailed search for an specific information (Vinson, 1999). This practice avoids the duplication of UX information or artifacts.

Agile teams can define templates to *arrange the UX-related information in the virtual tools*. Given the difference in the tools used by the ones involved in the team, we believe that the merge between UX and agile should not be in a tool level, but in a link level.

Agile teams should have an *awareness about the navigational distance* they could introduce between US and UX information. We believe that navigational distance are not a problem if they are kept at manageable levels. Given US structure (Cohn, 2004), placing UX information outside it is expected, but the relationship among the information in virtual environments should not fall into the Infinite distance type. Besides, this awareness can contribute to avoid duplication of UX information and facilitates information management and retrieval (Soares et al., 2015).

7 CONCLUSION AND STUDY LIMITATIONS

This paper presented an investigation on how USs and UX information are connected in agile virtual environments. As a result we presented some navigational distances types, which can bring problems to agile practitioners on retrieving and finding UX information. We also suggest recommendations to agile practitioners rethink the organization of the artifacts and documents in virtual environments.

Although the present case study has been done in the limited context of one organization, the artifacts and documents we explored are often used by agile practitioners (Garcia et al., 2019). Additionally, the theories and processes applied in our study were not based in a specific context, allowing such methods to be applied in other scenarios. We also believe that the results can be replicated in similar contexts, although adjustments may be required to better serve the particularities of each project.

As future work, we intend to conduct an interview with agile practitioners to gather feedback about our recommendations. Based on the recommendations presented in this study, a structure pattern for UX information arrangement, and its relationship to USs, is being developed by the authors.

ACKNOWLEDGMENT

We thank the financial support of the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001 and grant #2017/03397-0, São Paulo Research Foundation (FAPESP). We also thank the company for providing the documents for this study.

REFERENCES

Akman, S., Ozmut, M., Aydın, B., and Gokturk, S. (2015). Experience report: implementing requirement traceability throughout the software development life cycle. *Journal of Software: Evolution and Process*, 28(11):950–954.

Bjarnason, E. and Sharp, H. (2017). The role of distances in requirements communication: a case study. *Requirements Engineering*, 22(1):1–26.

Bohm, A. (2004). Theoreticai coding: Text analysis in grounded theory. *A companion to qualitative research*, 270.

Brhel, M., Meth, H., Maedche, A., and Werder, K. (2015). Exploring principles of user-centered agile software development: A literature review. *Information and software technology*, 61:163–181.

Choma, J., Zaina, L. A., and Beraldo, D. (2016). Userx story: incorporating ux aspects into user stories elaboration. In *International Conference on Human-Computer Interaction*, pages 131–140. Springer.

Cohn, M. (2004). *User stories applied: For agile software development*. Addison-Wesley Professional.

Da Silva, T. S., Martin, A., Maurer, F., and Silveira, M. (2011). User-centered design and agile methods: a systematic review. In *2011 Agile Conference*, pages 77–86. IEEE.

Da Silva, T. S., Silveira, M. S., Maurer, F., and Silveira, F. F. (2018). The evolution of agile uxd. *Information and Software Technology*, 102:1–5.

Garcia, A., da Silva, T. S., and Silveira, M. S. (2019). Artifact-facilitated communication in agile user-centered design. In *International Conference on Agile Software Development*, pages 102–118. Springer.

Garcia, A., Silva da Silva, T., and Selbach Silveira, M. (2017). Artifacts for agile user-centered design: a systematic mapping. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

Garrett, J. J. (2010). *The elements of user experience: user-centered design for the web and beyond*. Pearson Education.

Gregory, P., Barroca, L., Sharp, H., Deshpande, A., and Taylor, K. (2016). The challenges that challenge: Engaging with agile practitioners' concerns. *Information and Software Technology*, 77:92–104.

Harris, J. M., Hirst, J. L., and Mossinghoff, M. J. (2008). *Combinatorics and graph theory*, volume 2. Springer.

Hassenzahl, M. and Tractinsky, N. (2006). User experience-a research agenda. *Behaviour & information technology*, 25(2):91–97.

Hess, A., Diebold, P., and Seyff, N. (2017). Towards requirements communication and documentation guidelines for agile teams. In *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*, pages 415–418. IEEE.

Kashfi, P., Nilsson, A., and Feldt, R. (2017). Integrating user experience practices into software development processes: implications of the ux characteristics. *PeerJ Computer Science*, 3:e130.

Kassab, M. (2014). An empirical study on the requirements engineering practices for agile software development. In *2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 254–261. IEEE.

Liskin, O. (2015). How artifacts support and impede requirements communication. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 132–147. Springer.

Schön, E.-M., Winter, D., Escalona, M. J., and Thomaschewski, J. (2017). Key challenges in agile requirements engineering. In *International Conference on Agile Software Development*, pages 37–51. Springer, Cham.

Soares, H. F., Alves, N. S. R., Mendes, T. S., Mendonça, M., and Spínola, R. O. (2015). Investigating the link between user stories and documentation debt on software projects. In *2015 12th International Conference on Information Technology - New Generations*, pages 385–390.

Trienekens, J., Kusters, R., Himawan, H. B., and van Moll, J. (2018). Customer involvement in the scaled agile framework. In *Proceedings of the 20th International Conference on Enterprise Information Systems - Volume 2: ICEIS,*, pages 104–110. INSTICC, SciTePress.

Vinson, N. G. (1999). Design guidelines for landmarks to support navigation in virtual environments. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 278–285. ACM.