CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Usability Study of a Traceability Management Tool

Bachelor of Science Thesis in Software Engineering and Management

Per Skytt
Tobias Nersing

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

CHALMERS
UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

Usability Study of a Traceability Management Tool

Per Skytt
Tobias Nersing

Supervisor: Salome Maro
Examiner: Jennifer Horkoff

University of Gothenburg
Chalmers University of Technology
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Usability Study of a Traceability Management Tool

Per Skytt
Göteborgs Universitet
Gothenburg, Sweden
perskytt@outlook.com

Tobias Nersing
Göteborgs Universitet
Gothenburg, Sweden
tobsson@gmail.com

Abstract

This study aims to add to the body of usability research especially in regards to software tools. There is a lack of consideration to usability and empirical evaluation of software tools for developers in general, and in the field of traceability in context of usability is close to non-existent. Usability studies have been using various methods and measurements over the years but most of the time previous work in the field is not built upon and measurements and questionnaires are seldom fully revealed making reproduction and comparison across studies very difficult. This study evaluated the usability of a traceability management tool called Capra using a remote usability testing method where screen recording and post-test questionnaire are the means of gathering data. This study aims to build upon previous work by using validated and proven methods to assess usability and classify usability problems found to suggest improvement to the Capra-tool along with the aim to evaluate the overall usability.

Keywords

Capra, Traceability Management, Usability

I. Introduction

Usability in the software industry is a key issue which concerns the user experience. Today the interactive software market is constantly growing in the number of software tools, products and solutions being provided. The competition is hard as well as user-expectations growing standards of constantly anticipating software which is easier to handle, easier to understand and have increased quality over previous iterations. The usability of the software affects the user experience and in order to improve this usability, and thus the user experience, developers must understand and predict user behaviours. Toleman and Welsh [1] mentions that the typical design and development models used for software development tools ignores empirical user testing.

Usability is defined by ISO 9241 as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”. Good usability means that users will both enjoy the work they do more and be more efficient in doing it.

This thesis will be done in collaboration with the Capra Traceability Tool project [2], which is a result of the ITEA-funded project Amalthea4Public [3], where we will conduct a usability study on the Capra traceability management tool.

The current research of usability of traceability tools in the software engineering domain is quite lacking. The traceability tools and methods discussed in several papers lack empirical validation. Much of usability studies conducted in the HCI domain does not build upon previous work and this leads to increasing difficulty of comparing results and reproducing experiments [4].

To improve the result as well as add to the body of research in usability testing of traceability tools we will use standardized measurements of both objective and subjective data that have been proven to work and been validated in earlier research. By using validated tools and methods in this user study the benefit to the current body of research is provided by making our contribution reproducible and building upon previous work within the field- This study will aim show that standardized measurements and questionnaires can be used to evaluate the usability of a traceability management tool in an effective way and make the study both reusable and comparable to other similar studies. The data gathered from the study will be used to provide concrete suggestions about improvements regarding usability in the open source software Capra.

II. List of abbreviations

CUP - "Classification of Usability Problems"
HCI - "Human Computer Interaction"
HVAC - "Heating Ventilation Air-Conditioning"
IDE - "Integrated Development Environment"
SUS - "System Usability Scale"
SUT - "System Under Test"
UP - "Usability Problem"

III. Research Questions

Our thesis will aim to answer the following research questions:

- **RQ1:*How usable is the traceability management tool?***
  - SQ1 *How satisfying is the traceability management tool to use?*
- **RQ2: *What changes could be done to improve usability of the traceability management tool ?***
  - SQ1 *How can the efficiency be improved for the traceability management tool?*
  - SQ2 *How can we improve satisfaction of using the traceability management tool?*

The main objective of this study is to evaluate the current state of Capra from point of view of usability along with proposing possible improvements.

RQ1 asks the question of the usability of the traceability tool which we have clarified with an extra sub-question about satisfaction. RQ2 is about what kind of improvements this study and research will be able to suggest for the traceability tool. Sub-questions for RQ2 separates efficiency and satisfaction..

IV. Literature Review

In literature review by Nair et al., published in 2013 [5], the authors aimed to provide insight in how traceability research evolved in the Requirements Engineering conference during the previous 20 years. The authors wrote 4 suggestions for future research in the field and 2 of them included “tool qualification must be studied in more depth” and that “it is necessary to focus on the opinion and experiences of practitioners different to the researchers”. This is further substantiated in the systematic review and included case study by Torkar et al., 2012 [6] where they also studied requirements traceability, they claimed that most papers in their review focused on new features and extensions for tools and lacked validation. The authors writes that “most techniques and tools were not validated empirically.”

A study of software development tools used for refactoring by Mealy et al., 2007 [7] mentions the frequent lack of consideration regarding issues of usability, both in production and research, for software tools in general, and especially for software tools used for refactoring where the authors mentions that little research has been conducted on usability. While the study by Mealy et al focus on determining usability of a software tool used by developers for refactoring, our study will use a traceability management tool. In the systematic evaluation by Toleman & Welsh, 1998 [1] they studied design choices of software development tools and claimed that the general model for design and development used by the tool designers ignored empirical user testing. In the article by Đukić, et al. from 2015 [8] the authors talk about the current lack of or rather non existing focus on usability and traceability in information software systems.

In an article by Laura Faulkner from 2003 [9] it was found that using a low number of users in usability testing can give unreliable results in terms of how many problems were found. 5 users found as little as 55% of the problems, at 10 users it was 80% and at 20 users 95% of all the problems were found.

V. Methodology

A. Background

The methods used to evaluate usability generally fall into two categories: usability inspection methods and usability studies. Usability inspection methods involves evaluators who inspect a system in order to evaluate the usability which requires the evaluator to have some experience in usability [10]. Example of inspection methods are heuristic evaluations and cognitive walkthroughs. A usability study on the other hand is user-centered and usability test are used to assess the suitability of a system in regards to its intended purpose and intended users by revealing problems experienced by the user when trying to accomplish a set of tasks.

Remote usability test, also called unmoderated usability test, involves the user running the actual test on their own. The user is often in their home or office and their behaviours and interactions with the system is then captured using tools for example: screen-recordings, number of clicks, verbalization,eye movement, etc. This data can then later be analyzed. The method can be synchronous, the test is done and observed in real time by the tester or asynchronous, as in the user runs the test in their own time without and the testers receive the data for analysis afterwards.

This study will measure subjective usability using the questionnaire System Usability Scale created by John Brooke [11]. This is a Likert scale, it is simple and consists of ten items which can be used to get the subjective overall view of a systems usablility. SUS is generally used directly after a system has been used to capture the immediate response of the user. The SUS score is a single number and the process of getting to this number is to start by calculating the sum of the score contributions, values ranging from 0 to 4, from each item. Items 1,3,5,7,and 9 has a score contribution of the scale position minus 1, and items 2,4,6,8 and 10 has a score contribution of 5 minus the scale position. When you have the sum it will be multiplied by 2.5 to obtain the overall value.

Classification of usability problems scheme is a framework for describing and detailing usability problems. Its divided in two parts, pre-CUP and pos-CUP, where pre-CUP contains nine attributes which describes the UPs found in usability testing. The pre-CUP is presented to the developers which can then fill in the four attributes in post-CUP [12], [13].

B. Methodology

Before the tests were sent out twelve users had signed up to participate in the study. Out of these twelve only six actually completed the test giving us limited data to work with and one of our identified risks became a reality. These participants come from both academia and industry. A sample size of at least 10 users was the goal to reliably find most of the problems with usability [9]. To find participants from the academic domain we posted requests for participants via the facebook-community pages for the software engineering program and management bachelor's program asking students from second and third year. To find participants from the industry personal contacts were used.

We categorized the data to be collected into two main groups: objective data and subjective data. Subjective data consists of measurements that concern users' perception of or attitudes towards the interface, the interaction, or the outcome whereas objective data is concerned with data not dependent on the user perception or attitudes. Studying both subjective and objective measurements can, as Hornbaek [4] points out, show different results regarding usability of an artifact. We will besides keeping the types separate include both types of data in this study as we believe a more complete picture of usability might be achieved by doing so.

The study was conducted as a remote usability study, this is a method which means that those conducting the study and the participants are not in the same room or location as the test participants. Our remote usability method was of the asynchronous variant, where evaluators of the test won't be interacting or gathering data in real-time while the participants perform the test. We chose this method based on the benefits of the test being location-independent, time saving and easy to scale for a large sample while still considered effective and suitable on a low-budget [14]. The asynchronous method will not be able to record observations of the user or spontaneous verbalizations, although the synchronous method can be perceived as more intrusive [15] and would require more coordination in regards of time and schedule. Another benefit of the remote usability testing method is that the participants will use the software in their own environment.

To record these sessions Open Broadcaster Software [16] usually referred to as OBS was used. This software was chosen because it is free to use, it's open source and works on Windows, Mac and Linux. This made it possible to create a set of instructions that worked for every user and the researcher only needed to learn 1 software in case the users needed help with the installation or usage of the recording. To upload the recordings Google Drive [17] was used. To make sure that there was enough space on the account to upload all the recordings the researchers created test recordings to see how big on average a file would be. It was found that the recordings were small enough that even if every user took twice as long as the expected time the free space would be enough.

The participants of the test received all the material needed to understand the goals of the study in a PDF document. This document can be seen in the appendix section A This material consisted of instructions to install needed software for gathering data, install and run the. Capra tool, an introductory video of Capra and specific instructions for how to start the test and upload the data. The participants was also provided with 12 tasks which we estimated wouldn't take more than 30 minutes on average to complete. To complete these tasks an example project of a Heating, ventilation and air conditioning (HVAC) system was provided in which included among other files: statecharts, requirements,jUnit tests and a feature-model which were used in the tests. A SUS questionnaire [11] to fill out post-test.

The 12 tasks that users were asked to complete are these:

1) Create a trace link between the Requirement 4 and the ITOS feature.
2) Create a trace link between ITOS feature and TemperatureAdapter Statechart.
3) Create a trace link between TemperatureAdapter Statechart and the ITOSTest java class.
4) View the trace links of the ITOS feature through PlantUML diagram.
5) Use transitivity-function to see the whole trace of connections in relation to the ITOS feature.
6) Remove the trace link between ITOS feature and TemperatureAdapter Statechart.
7) Delete the ITOSTest java class.
8) Open the Eclipse Problem view. Find the warning concerning the deleted ITOSTest java class and use the Eclipse "Quick Fix" function to remove all affected trace links.
9) Create a trace link between Requirement 4 and HVAC_manager feature.
10) Create a trace link between HVAC_manager feature and TemperatureAdapter Statechart.
11) View the trace links of HVAC_manager feature through PlantUML diagram.
12) View the Capra traceability matrix of HVAC_manager feature, TemperatureAdapter Statechart and ITOS feature to make sure ITOS feature isn't linked to the other two.

An expert on Capra, a developer from the project, was asked to run the scenarios to establish a best-case of time, error rate, success/failure-rate for the intended user-scenarios in the user test, which was used as a baseline for our objective measurements.

We intend to use post-test measures for perceived usability as a subjective measure, here in form of the validated questionnaire System Usability Scale [11]. To be able to get more information about what the participants think of the different parts of the software additional questions have been added. These questions are linked to the different tasks the participants perform asking about specific features of the software. These additional questions was not part of the overall SUS-score but only used a 5 point likert-scale ranging from strongly disagree to strongly agree. To end the questionnaire there are two free form questions where the participants are asked to write down any improvements or new features they want to see in the software. The questionnaire that users will be asked to fill out can be seen in table **??** "Questionnaire".

As mentioned in Hornbaek's paper [4] there appears to be a lack of studies using validated instruments for measuring satisfaction which builds upon previous work and we intend to not further add to this type of disarrangement to enable future comparison to related studies and higher reproducibility of our study.

C. Data Analysis

The authors examined the answers in the questionnaires and the answers regarding specific functions as well as the overall SUS-score to get an initial idea of the usability issues which could be present in the Capra software. Both authors watched each screen-recording separately to identify usability problems(UP). The initial criteria we used for identification of UPs derived from video analysis was based on Nielsen's 10 principles [18] and the definition for usability problems described by nine problem criteria by Jacobsen et al.(1998) [19]:

1) The user articulates a goal and cannot succeed in attaining it within three minutes
2) The user explicitly gives up
3) The user articulates a goal and has to try three or more actions to find a solution
4) The user produces a result different from the task given
5) The user expresses surprise
6) The user expresses some negative affect or says something is a problem
7) The user makes a design suggestion
8) The evaluator generalizes a group of previously detected problems into a new problem

The Usability problems(UPs) were then documented in a spreadsheet by each author separately detailing the following information:

- A headline that summarizes the problem
- An explanation that details the problem - *As many details as possible and to ensure that the description was understandable without knowledge of the test sessions or the videos*
- A description of why the problem is serious to some or all users of the software - *For example if users get confused, express that they are insecure, or cannot finish their tasks*
- A description of the context - *A description of the context where the problem was identified, for example in a certain task scenario or part of the user interface*
- Identified in task
- Participant ID - *Which test participants session was the UP identified in*
- Which evaluator found this

To see a list of all identified UPs and detailed information about each of them refer to the appendix section A.

The objective data consists of screen recordings where the users work on the scenarios/tasks they are given during the user test. We will measure time spent on tasks, number of errors and the success/failure to complete a task. In the event of a crash of Eclipse or other error halting the user test, if the error is not deemed related to the SUT(software under test), the time to recover from this error is not counted in the total task time.

The time for task one is counted from when the user starts the test until the goal of task one is completed. When the goal for task one is completed the clock starts counting the time for task two. This way of counting the time is the same for all tasks. If a user fails to complete a task or gives up trying to complete a task the time is stopped when the user appears to start the next task, usually this was seen by the user going to a specific part of the interface or looking at the instructions.

Errors that was looked for was either when an error message appeared or if an action did not result in what was expected i.e. a bug. The success and failure rate of the tasks were 100% if the user completed the goal of the task. No consideration to how many steps or how long time the user took was taken here. If the user skipped or forgot to do a task it was 0%. If the user tried to complete the task but didn't succeed, depending on how close they came, an approximation was done to get a percentage somewhere between 0 and 100 to reflect how close the user was to complete the task.

These UPs were then discussed and compared and organized in a consolidated list which both authors agreed upon where duplicates were marked as one UP resulting in a list of UPs to be classified using Classification of Usability Problems Scheme(CUP) [12], [13]. This list would contain all the agreed upon UPs as well as reference to relevant tasks and screen-recordings.

These UPs were then classified using Classification of Usability Problems Scheme. We divided the UPs between the two authors, classifying half of the UPs each, and then reviewed each others classifications and through discussion made edits and agreed upon the CUP classifications. All the classifications can be seen in full in the appendix section A.

VI. Results

This study got responses from 6 participants of the user test where each answered the questionnaire and sent in a screen-recording made during their run of the tasks in the test. The participants consisted of 83,3% 3rd year bachelor students of software engineering and 16,7% industry practitioners according to responses of the first question about the participant background. On a likert scale ranging from "1", strongly disagree, to "5", strongly agree, the participants were asked two more questions about their background. We asked how familiar they are with the concept of traceability in software engineering. Out of all participants 66,7% answered "1", strongly disagree, and 16,7% answered "2" and 16,7% answered "3" on the scale in regards to their familiarity of the concept of traceability in software engineering. The last background question using the same likert scale asked how familiar the participants were with the Eclipse IDE. 33% of the particpants answered "2" and 66,7% answered "3". Although a majority of the participants weren't familiar with the concept of traceability, most were familiar with the Eclipse IDE.

A. Video Data

Together the authors identified a total of 27 UPs during their individual evaluation of the screen-recordings, 13 UPs and 14 UPs respectively. As can be seen in table **??**, out of these 16, 7 were classified as minor, 6 as moderate and 3 as severe. During the tests no bugs were encountered and the only other error message that appeared, which occurred once to only one participant, was ruled to not have anything to do with Capra and only one of the users experienced this error and is therefore not considered in the analysis.

The average time per task compared to the expert times can be seen in table **??**. The expert is faster at every task except 1, the expert is being slower here but note that that the expert had the example project's organization of files slightly different than the example project for the actual user test.

The tasks 1,4,5,8 in the table has a difference of more than 1 minute of time for the participants in the study to complete the task compared to the expert's time, and task 12 stands out with over 2 minutes more time elapsed on average for the participants of the test compared to the expert. The tasks 1,4,5,8 and 12 relates to the features of creating trace link, PlantUML-visualization, Transitivity, warnings and Trace Matrix-visualization respectively.

The percentage of how much of a task the participants completed can be seen in table **??**. The only task where any participant failed to complete any part of the task was in task 6 where 2 participants had a completed percentage of 0%. All other participants managed to complete the same task in full. Another task where the participants were struggling was task 12. It took the users a long time to complete this task but 4 out of 6 participants managed to complete the task and 2 participants managed to complete 75% of the task.

B. Questionnaire Data

The questionnaire responses for the first 10 questions following the background of the participants were based on SUS and in the following table is the SUS-score from each participants responses along with overall average and median. The average SUS-score for Capra in this study is 41,25

Participant	SUS-Score
1	40
2	32,5
3	37,5
4	35
5	57,5
6	45
Total Average	41,25
Total Median	38,75

After the ten SUS questions there were six additional questions about certain features in Capra. The users were asked to use a likert scale from 1-5, strongly disagree to strongly agree to these questions:

1) *"I thought the removing trace links-feature in the Capra tool was easy to use"*
2) *"I found the creating links-feature of the Capra tool very cumbersome to use"*
3) *"I thought the "Trace Matrix" in the Capra tool was easy to use"*
4) *"I thought the "PlantUML View" in the Capra tool was easy to use"*
5) *"I found that the "Transitivity"-function in the Capra tool was well integrated"*
6) *"I found the notification/warning functions in the Capra tool were well integrated"*

In the table **??** additional questions each of the result from the additional questions about certain features of Capra, the result showed in the table contains the average, median, related feature and a percentages of of all responses to each question.

In the last section of the questionnaire the participants could answer two free-form questions: "What kind of improvements would you suggest for the Capra tool?" and "What kind of additional features would you suggest for the Capra tool?". To the question "What kind of improvements would you suggest for the Capra tool?" only 4 of the 6 participants responded:

- *"Too many dependencies on other plugins made the getting started phase very long. I'd prefer to have them integrated."*
- *"Trace matrix should have limits on the length of names/info in each cell"*
- *"Better guide to get started and 1 installation package instead of the X amount."*
- *"To make remove trace function more obvious (as long as I couldn't fine it). "*

To the question "What kind of additional features would you suggest for the Capra tool?" only 3 out of the 6 participants responded:

- *"I missed the possibility to interact with the trace through the visualizations, such as PlantUML View and the matrix"*
- *"When showing tooltips (hovering over icons) offer some more information"*
- *"I'm not that familiar with the topic. I think it would probably nice to have some small tutorial/tips about some functionalities."*

VII. Discussion

From the data gathered and the free text questions these are some of the more interesting findings. Two users suggested that the installation of Capra requires too many dependencies making the process taking more time than if dependencies were integrated into Capra. The example project used for the test had the need for some additional dependencies which wasn't related to Capra but we suspect this could have been misinterpreted by the users as part of the installation of Capra although clearly divided and explained to the test participant this could be the case.

Furthermore the additional questions about certain feature the participants answered reveals that the functionality of removing trace links could be improved, since most of the users on average selected strongly disagree that this feature is easy to use. And looking at our results from analyzing the videos the data shows that task 6 which involved removing a trace link during the test had a completion rate of only 66,67% whereas all other tasks had a completion rate of >90%. Although task 6 had an average time-per-task of 1 minute and 41 seconds the low completion rate together with the fact that 4 users completed the task successfully and 2 of the users completed 0% of the task involving removal of a trace link this feature's usability issues should be deemed a high priority to fix.

The "Traceability Matrix" feature of Capra was used in task 12 of the test which had the highest average task completion time of 3 minutes and 37 seconds and only 4 of the users managed to complete the task. The question in the questionnaire about the matrix feature's ease of use had an average of 2,5 on the likert scale but during our analysis we found that when the user wants to see a matrix of artifacts of different types the selection of said artifacts can be problematic and this usability issue is detailed in CUP with identifier "UP16" in the appendix. We suggest this UP be deemed severe to improve the efficiency and performance of the Capra tool.

Research Question 1 We first asked the question of how usable the Capra traceability management tool is and how satisfying Capra is to use. The overall usability of Capra according to the System Usability Scale shows that Capra in this study received a average score of 42,5 out of 100 which is is regarded a low score since the average is somewhere around 68 [20].

Sub Research Question 1 On a 5 point scale only 1 user rated a 3 to the question if they would like to use Capra frequently, all other users gave a rating of 1 or 2. This together with the fact that no user rated higher than 2 on the question if they felt confident using the system shows that the satisfaction of using Capra is low.

Research Question 2 The second research question we asked was regarding which improvements could be made for the Capra tool. We asked the participants in the user test to suggest improvements in a free form text in the questionnaire where the answers, as seen in the result-section, were related to three aspects: the installation, the matrix-feature and remove trace function.

Sub Research Question 1 The efficiency of Capra appears to be high for several features. Creating trace links, visualizing features such as transitivity and viewing a diagram and using the problem view to automatically fix errors were features that the participants had few issues with and were completed quickly. The biggest concerns when it comes to efficiency are removing trace links and using the matrix to visualize links between artifacts. The participants had problems completing these tasks and the ones that managed took a very long time to do it.

To remove trace links the user is required leave the project that they are working in and look in the new project where Capra stores its files. This is inconsistent with how all other features of Capra work and it requires a lot more clicks than for example creating a link. We suggest that this feature is made available through less steps and in line with how other features of Capra work either by allowing the user to remove links similarly to how they create a link or allow the user to interact and edit links when they visualize them.

When using the matrix view the users appeared confused about how to add artifacts. To accomplish the task we asked them to they had to find a specific file in which all artifacts could be accessed at the same time. To make this easier for the user allowing artifacts from several different places to be added would help.

Sub Research Question 2 We believe the low satisfaction of using Capra comes from a few features dragging down the overall experience. Improving the efficiency would likely help the satisfaction. We suggest giving the users more information about what the different elements of the UI does. Adding tool tips when hovering over certain icons to tell the users what it does or can do.

VIII. Conclusions

The main objective of this study was to evaluate the current state of Capra from the point of view of usability along with proposing possible improvements. Our goal going forth with this study was to build upon previous work within usability engineering by using a validated questionnaire, SUS, validated classification of usability problems, CUP and also adding to the usability research field of traceability tools with a study that is reproducible. We showed that we could measure the usability and suggest improvement for the Capra tool.

We designed a Remote Usability Test where we gathered enough data to identify and classify 16 usability problems. There were only 6 participants that actually finished the test which limits the extent to which you can draw conclusions. It was also somewhat of a convenience sample that also could have an unwanted impact on the result.

Even though the limitations of the study might threaten the validity of the result we believe it reflects the reality but to prove this further doing the study again with a larger sample and a more mixed background would be appropriate.

IX. Future Work

Future work we suggest is firstly to reproduce this study after implementation of improvements with a larger sample size and perhaps an implementation of think-a-loud protocol where verbalizations of the users during the test are recorded as well. Secondly software tools for developers in general would benefit from an increased consideration and application of usability studies.

X. Acknowledgements

Both authors would like to thank Salome Maro for contributing guidance and support.

XI. Threats To Validity

A. Internal Validity

The initially identified possible risks during this study included not getting a large enough sample, a sample of at least 10 participants. The validity of the data gathered being able to identify >80% of all usability problems requires only 5 participants according to Nielsen, although this number has been disputed and research indicates that 10 participants are needed to identify around 80% or more of the usability problems. Our sample size started at 12 participants who confirmed their participation and started the test but only 6 of the participants actually finished the test and sent in the resulting data which can impact the number of usability issues detected in this study.

Another issue which effects our data collected in regards to validity is that several of the 6 users did not follow the instructions of the test accurately regarding screen-recordings. The instruction was given to include all displays used during the test in the video-recordings sent to us but this was not the case. All users recorded the screen containing Eclipse IDE and the SUT. This meant that we could not determine if the users was seeking help or doing other work during the test when users interacted on a second display not recorded.

We did not implement a think-aloud-protocol for the test, the protocol where users would have recorded their voices and be encouraged to speak out all their thoughts during test-session. The absence of such think-aloud-protocol during the test made it impossible for us to accurately determine “instances of frustration” of the attribute "Impact" in the CUP classification and the protocol’s implementation in this study would have generated data which could further support identification of user patterns. The data from the voice recordings could reveal the users’ thoughts on their various interactions with Capra and back our claims of usability problems along with revealing more usability issues.

References

[1] M. A. Toleman and J. Welsh, “Systematic evaluation of design choices for software development tools,” *Software-Concepts & Tools*, vol. 19, no. 3, pp. 109–121, 1998.

[2] S. Swart. Capra. Accessed: 2017-03-23. [Online]. Available: http://projects.eclipse.org/

[3] A. project. Amalthea. Accessed: 2017-03-23. [Online]. Available: http://amalthea-project.org/

[4] K. Hornbæk, “Current practice in measuring usability: Challenges to usability studies and research,” *International journal of human-computer studies*, vol. 64, no. 2, pp. 79–102, 2006.

[5] S. Nair, J. L. De La Vara, and S. Sen, “A review of traceability research at the requirements engineering conference re@ 21,” in *Requirements Engineering Conference (RE), 2013 21st IEEE International*. IEEE, 2013, pp. 222–229.

[6] R. Torkar, T. Gorschek, R. Feldt, M. Svahnberg, U. A. Raja, and K. Kamran, “Requirements traceability: a systematic review and industry case study,” *International Journal of Software Engineering and Knowledge Engineering*, vol. 22, no. 03, pp. 385–433, 2012.

[7] E. Mealy, D. Carrington, P. Strooper, and P. Wyeth, “Improving usability of software refactoring tools,” in *Software Engineering Conference, 2007. ASWEC 2007. 18th Australian*. IEEE, 2007, pp. 307–318.

[8] V. Đukić, I. Luković, M. Črepinšek, T. Kosar, and M. Mernik, “Information system software development with support for application traceability,” in *International Conference on Product-Focused Software Process Improvement*. Springer, 2015, pp. 513–527.

[9] L. Faulkner, “Beyond the five-user assumption: Benefits of increased sample sizes in usability testing,” *Behavior Research Methods, Instruments, & Computers*, vol. 35, no. 3, pp. 379–383, 2003.

[10] J. Nielsen, “Usability inspection methods,” in *Conference companion on Human factors in computing systems*. ACM, 1995, pp. 377–378.

[11] J. Brooke *et al.*, “Sus-a quick and dirty usability scale,” *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.

[12] E. T. Hvannberg and E. L.-C. Law, "Classification of usability problems (cup) scheme." in *Interact*. Citeseer, 2003.

[13] S. G. Vilbergsdóttir, E. T. Hvannberg, and E. L.-C. Law, "Classification of usability problems (cup) scheme: augmentation and exploitation," in *Proceedings of the 4th Nordic conference on Human-computer interaction: changing roles*. ACM, 2006, pp. 281–290.

[14] E. McFadden, D. R. Hager, C. J. Elie, and J. M. Blackwell, "Remote usability evaluation: Overview and case studies," *International journal of human-computer interaction*, vol. 14, no. 3-4, pp. 489–502, 2002.

[15] N. Ghasemifard, M. Shamsi, A. R. R. Kenari, and V. Ahmadi, "A new view at usability test methods of interfaces for human computer interaction," *Global Journal of Computer Science and Technology*, vol. 15, no. 1, 2015.

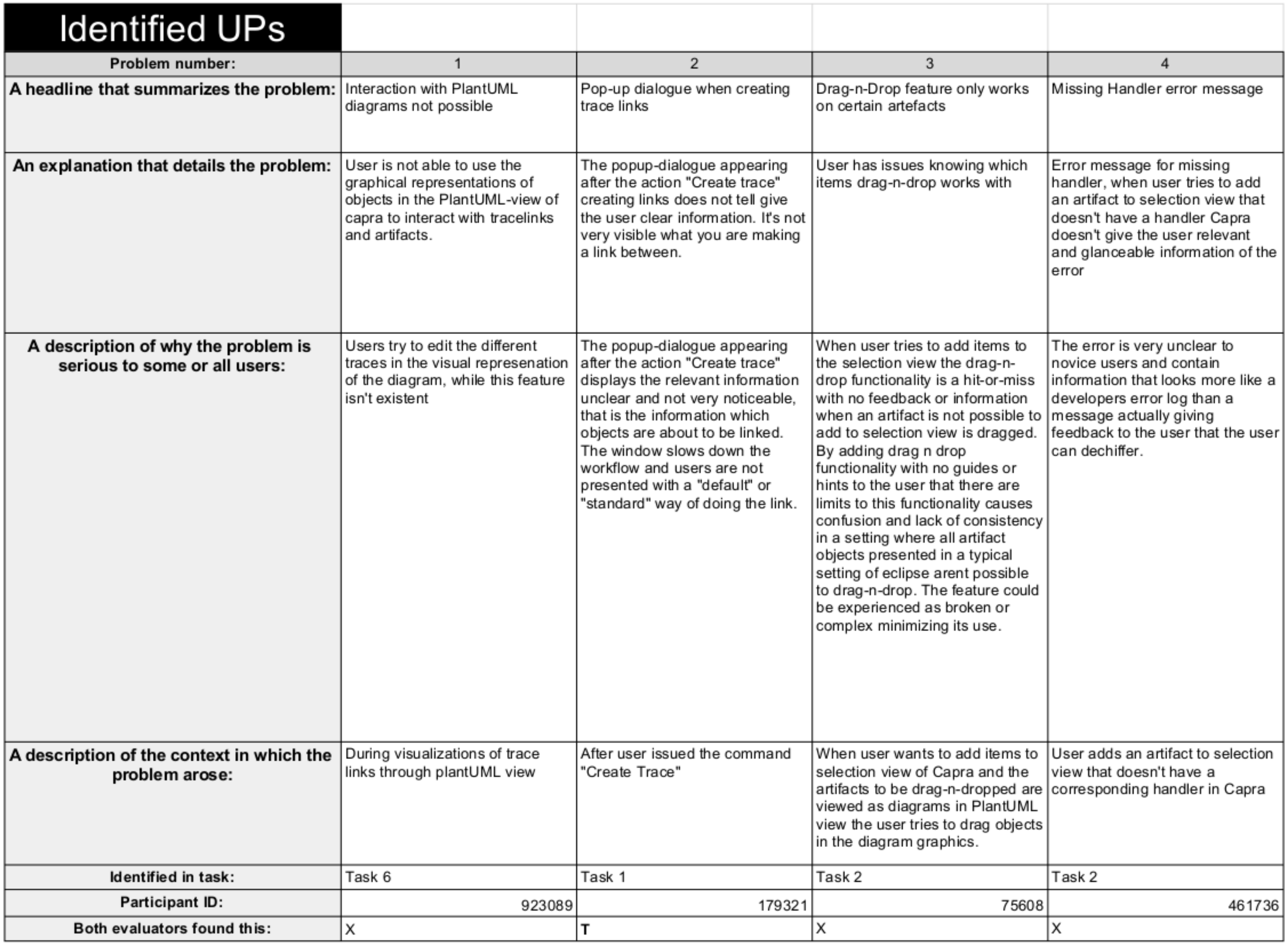
[16] Open boradcaster software. Accessed: 2017-07-05. [Online]. Available: https://obsproject.com/

[17] Google drive. Accessed: 2017-07-05. [Online]. Available: https://www.google.com/drive/

[18] J. Nielsen, "10 heuristics for user interface design: Article by jakob nielsen," 1995, accessed: 2017-03-23. [Online]. Available: https://www.nngroup.com/articles/ten-usability-heuristics/

[19] N. E. Jacobsen, M. Hertzum, and B. E. John, "The evaluator effect in usability studies: Problem detection and severity judgments," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 42, no. 19. SAGE Publications, 1998, pp. 1336–1340.

[20] J. Sauro. Measuringu: 10 things to know about the system usability scale (sus). Accessed: 2017-05-23. [Online]. Available: https://measuringu.com/10-things-sus/



Identified UPs				
Problem number:	1	2	3	4
A headline that summarizes the problem:	Interaction with PlantUML diagrams not possible	Pop-up dialogue when creating trace links	Drag-n-Drop feature only works on certain artefacts	Missing Handler error message
An explanation that details the problem:	User is not able to use the graphical representations of objects in the PlantUML-view of capra to interact with tracelinks and artifacts.	The popup-dialogue appearing after the action "Create trace" creating links does not tell give the user clear information. It's not very visible what you are making a link between.	User has issues knowing which items drag-n-drop works with	Error message for missing handler, when user tries to add an artifact to selection view that doesn't have a handler Capra doesn't give the user relevant and glanceable information of the error
A description of why the problem is serious to some or all users:	Users try to edit the different traces in the visual represenation of the diagram, while this feature isn't existent	The popup-dialogue appearing after the action "Create trace" displays the relevant information unclear and not very noticeable, that is the information which objects are about to be linked. The window slows down the workflow and users are not presented with a "default" or "standard" way of doing the link.	When user tries to add items to the selection view the drag-n-drop functionality is a hit-or-miss with no feedback or information when an artifact is not possible to add to selection view is dragged. By adding drag n drop functionality with no guides or hints to the user that there are limits to this functionality causes confusion and lack of consistency in a setting where all artifact objects presented in a typical setting of eclipse arent possible to drag-n-drop. The feature could be experienced as broken or complex minimizing its use.	The error is very unclear to novice users and contain information that looks more like a developers error log than a message actually giving feedback to the user that the user can dechiffer.
A description of the context in which the problem arose:	During visualizations of trace links through plantUML view	After user issued the command "Create Trace"	When user wants to add items to selection view of Capra and the artifacts to be drag-n-dropped are viewed as diagrams in PlantUML view the user tries to drag objects in the diagram graphics.	User adds an artifact to selection view that doesn't have a corresponding handler in Capra
Identified in task:	Task 6	Task 1	Task 2	Task 2
Participant ID:	923089	179321	75608	461736
Both evaluators found this:	X	T	X	X

5	6	7	8	9	10
Matrix selection using outline and diagram objects	Transitivity refresh	Transativity status	PlantUML diagram viewed through selectiong a trace in tracemodel instead of selecting actual artifact	Transitivity status hidden	Removing trace links does not autosave
The outline window in eclipse was used for selection along with a previous selection of an object in the diagram-editor which resulted in a matrix showing instead of Class diagram which wasn't intended by the user	No indication that you have to reload the artifact after toggling Transitivity.	No feedback or clue that Transitivity is enabled.	user saw a diagram through traceModel.xmi clicking on the trace object when user was suppose to click the single object ITOSFeature to get correct diagram	Visibility of the signifier for transativity function affordance is very low.	Removing a trace link from artifactModel.xmi requires the file to be saved manually to take effect.
The matrix-function is supposed to be viewed when two or more artifacts are selected, otherwise the plantUML diagram is shown. When users interact with the editors such as the .pld editor the user can click on a item in the diagram editor and later select another object in the outline view of eclipse and a selection of 2 artifacts are made but this is not clearly conveied to the user. The selection made by clicking on an object in the pld editor remains selected when using the outline to select objects. The selection of 2 artifacts to view a matrix where the 2 artifacts are in separate windows of the the project explorer, package explorer, different editors its not possible creating an inconsistency in the capabilities of the feature	The user receives no feedback that you actually need to manually refresh the PlantUML-diagram by clicking on the object once more after activating transitivity which isn't hinted to the user at all, but the user will "accidentally" notice transitivity of the links displaying when they try to solve the issue of not receiving instructions or feedback	Transitivity function is activated and then you have to enter the menu for transitivity to see if its ticked as activated, no more feedback, and can be very confusing that there isn't a visual cue but a "hidden" tick inside the menu	The PlantUMLdiagram-view by clicking on a trace in the traceModel.xmi isn't the same diagram as when the user clicks an object which is part of a trace link and this is unclear that it is different diagram than the PlantUML view of a single selected artifact, not selected trace link. Users dont get a mental model of the feature corresponding to the system models capablities	User need to go into a menu to check if transitivity is enabled or not.	Other parts of Capra does not require manual saving making the system inconsistent.
Using the matrix view and clicking on an object inside the pld diagram editor	Enabling the transativity function when viewing a PlantUML visualization and the transativity function is previously off.	When activating transativity-function	When users want to view the trace links in PlantUML view and navigates to the tracemodel and views the PLantUML diagrams of the traces instead of viewing the actual objects.	Any time the user wants to see the relationships between more than 2 artifacts.	User tries to remove a link that has already been created.
Task 4	Task 5	Task 5	Task 4	Task 5	Task 6
923089	75608	75608	923089		
P	X	T	P	X	X

11	12	13	14	15	16
Trouble finding where to remove trace links	Trying to remove trace links in the wrong place	Popup asking to save overwritten work gives no information.	Hard to see an overview of trace links that are created.	No indication if items in selection view needs to be selected or not.	Trouble finding artifacts for matrix view.
Users are unsuccessful in knowing where to look for removing a trace link.	User tries to edit trace-links through the Selection View.	Dialogue window asking user to discard changes made outside the editor (via quick-fix) confuses the user since the information given makes the consequences hard to understand (Eclipse pop-up)	User failed to remove a trace in a previous task, they thought the trace was created (but the users missed to create the link in first place), user therefore have issues when asked to delete the trace from the tracemodel because the user can't find clear information of which links are created.	User is uncertain if items in Selection View need to be selected or not to create a trace link.	Problem with selecting the three different objects to show matrix.
Removing a trace link is hidden behind many steps. Creating trace links is done by directly finding the artifacts to link and using them, but you cannot remove links by using the actual artifacts making the system inconsistent.	Adding and removing trace links have two very different approaches making the user unsure of how to actually remove trace links.	There is nothing in the popup indicating which data that will be affected forcing the user to backtrack to find what has gone wrong.	There is no way to easily check what trace links already exists. Even if the user knows where to look it's required to look at every link created individually to know what it's a link between.	Other than the name there is no indication that all items in Selection View will be used to create a trace link. The text in the popup saying what the trace link will be between is something that users overlook.	Creating a matrix from different files required a loot of looking around. The user cannot select the files from the Project Explorer view.
User wants to remove a trace link that has already been created but is unsuccesful in finding where to do this.	User wants to remove a trace link that has already been created.	User removes a trace link but doesn't save traceModel.xmi, after the quickfix through the problem view is applied the popup appears.	User wants to remove a trace link but cannot find where they are stored.	User wants to create a trace link between two artifacts.	User wants to create a matrix from different files.
Task 6	Task 6	Task 8			Task 12
	901605				
X	X	X	P	X	X

Attribute	Values		
Defect ID	UP1		
Frequency	4/6		
Trigger	Trying to remove trace link, task 6		
Context	When interacting with PlantUML visualization diagram		
Description	Users try to edit the traces in the visual representation of the diagram, while this feature isn't implemented.		
Defect Removal Activity	User test		
Impact	**Severity**		Moderate
	Task Efficiency(%/min)		39,60%
	Task Effectiveness	Completion Rate	66,67%
		Mean Time on Task(min:sec)	01:41
Failure qualifier	Missing		
Expected phase	Functional Requirements		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP2		
Frequency	6/6		
Trigger	When issuing "Create Trace"-command, Task 1, 2, 3, 9, 10		
Context	When user right-clicks in the selection view and clicks "Create trace" the popup-dialogue window appears		
Description	The popup-dialogue for creating links does not tell give the user clear information. It's not clear to the user what artifacts you are going to link.		
Defect Removal Activity	User test		
Impact (average calculations of all relevant tasks)	**Severity**		Minor
	Task Efficiency(%/min)		65,71%
	Task Effectiveness	Completion Rate	99,67%
		Mean Time on Task(min)	01:31
Failure qualifier	Overlooked		
Expected phase	Presentation design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP3		
Frequency	4/6		
Trigger	When user adds unsupported artifacts to selection view, Task 1, 2, 3, 9, 10		
Context	When adding artifacts by drag-and-drop from an editor's window (or package/project explorer to selection view		
Description	User has issues knowing which artifact types drag-n-drop functionality supports when dragging artifacts to selection view		
Defect Removal Activity	User test		
Impact (average calculations of all relevant tasks)	**Severity**		Moderate
	Task Efficiency(%/min)		65,71%
	Task Effectiveness	Completion Rate	99,67%
		Mean Time on Task(min)	01:31
Failure qualifier	Incongruent mental model		
Expected phase	Functional Requirements		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP4		
Frequency	2/6		
Trigger	User adds and artifact type which lacks a configured capra-handler, Task 2		
Context	When user want to add an artifact in a eclipse project to the selection view of capra		
Description	Error message regarding a missing handler when user tries to add artifact to selection doesn't give the user glanceable and clear information, too much information on screen confusing for some users.		
Defect Removal Activity	User test		
Impact	**Severity**		Minor
	Task Efficiency(%/min)		56,73%
	Task Effectiveness	Completion Rate	98,33%
		Mean Time on Task(min)	01:44
Failure qualifier	Irrelevant		
Expected phase	Presentation Design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP5		
Frequency	1/6		
Trigger	When selecting artifacts as objects inside a graphical pld editor and outline matrix view appears, Task 4		
Context			
Description	The outline window in eclipse was used for selection along with selecting an object in the diagram in the editor which resulted in a matrix showing instead of Class diagram which wasn't intended by the user. Capra doesn't allow selection for matrix between windows except outline and editor in this case which the selection inside the pld diagram is active but not noticed by the user causing a matrix to display instead of a diagram.		
Defect Removal Activity	User test		
Impact	**Severity**		Minor
	Task Efficiency(%/min)		43,60%
	Task Effectiveness	Completion Rate	96,67%
		Mean Time on Task(min)	02:13
Failure qualifier	Overlooked		
Expected phase			
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP6		
Frequency	6		
Trigger	When activating transitivity function for PlantUML, Task 5		
Context	In the PlantUML view of selected artifact when user toggles transitivity in this window through the Triangle-button		
Description	No indication that you have to reload the artifact after toggling Transitivity.		
Defect Removal Activity	User test		
Impact	**Severity**		Severe
	Task Efficiency(%/min)		72,83%
	Task Effectiveness	Completion Rate	98,33%
		Mean Time on Task(min)	01:21
Failure qualifier	Incongruent mental model		
Expected phase	Functional Requirements		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP7		
Frequency	6		
Trigger	Activating transitivity, Task 5		
Context	In the PlantUML view of selected artifact when user toggles transitivity in this window through the Triangle-button		
Description	No feedback or clue that Transitivity is enabled besides inside the dropdown menu used to activated it(which is hidden again once transitivity activated) , but there is no way for the user to know its activated besides refreshing the PlantUML-view or entering the same menu where it was activated.		
Defect Removal Activity	User test		
Impact	Severity		Moderate
	Task Efficiency(%/min)		72,83%
	Task Effectiveness	Completion Rate	98,33%
		Mean Time on Task(min)	01:21
Failure qualifier	Missing		
Expected phase	Presentation Design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP8		
Frequency	2/6		
Trigger	Task 4		
Context	When user clicks on a artifact to visualize a PlantUML diagram but selects a trace object inside tracemodel.xmi instead of the artifact part of a trace		
Description	User visualized a diagram through traceModel.xmi by clicking on the trace object when user was suppose to click the single object ITOSFeature to get asked for diagram in the task. The user isn't using the feature of PlantUML where the its suffice to select the object part of a trace but instead looks through the traces in tracemodel for diagram.		
Defect Removal Activity	User test		
Impact	Severity		Minor
	Task Efficiency(%/min)		43,60%
	Task Effectiveness	Completion Rate	96,67%
		Mean Time on Task(min)	02:13
Failure qualifier	Overlooked		
Expected phase	Presentation Design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	**Values**		
Defect ID	UP9		
Frequency	6/6		
Trigger	Use transitivity-function to see the whole trace of connections in relation to the ITOSfeature.		
Context	Any time the user wants to see the relationships between more than 2 artifacts. User is in the PlantUML view and press the triangle to activate transitivity.		
Description	Visibility of the signifier for transitivity function affordance is very low.		
Defect Removal Activity	User test		
Impact	**Severity**		Moderate
	Task Efficiency(%/min)		74%
	Task Effectiveness	Completion Rate	100%
		Mean Time on Task(min)	01:21
Failure qualifier	Incongruent mental model		
Expected phase	Presentation design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	**Values**		
Defect ID	UP10		
Frequency	4/6		
Trigger	Remove the trace link between ITOS feature and TemperatureAdapter Statechart.		
Context	User tries to remove a trace link that has already been created. User has traceModel.xmi open and finds and removes the correct trace link.		
Description	Removing a trace link from artifactModel.xmi requires the file to be saved manually to take effect. No indication is given to the user that this action is required.		
Defect Removal Activity	User test		
Impact	**Severity**		Severe
	Task Efficiency(%/min)		39.61%
	Task Effectiveness	Completion Rate	66,67%
		Mean Time on Task(min)	01:41
Failure qualifier	Incongruent mental model		
Expected phase	Functional requirements		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP11		
Frequency	6/6		
Trigger	Remove the trace link between ITOS feature and TemperatureAdapter Statechart.		
Context	User wants to remove a trace link that has already been created but is unsuccesful in finding where to do this. User is mainly looking in the project folder and not in the new folder created by Capra containing the links.		
Description	User is unsuccessful in knowing where to look for removing a trace link.		
Defect Removal Activity	User test		
Impact	**Severity**		Moderate
	Task Efficiency(%/min)		39.61%
	Task Effectiveness	Completion Rate	66,67%
		Mean Time on Task(min)	01:41
Failure qualifier	Overlooked		
Expected phase	Presentation design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

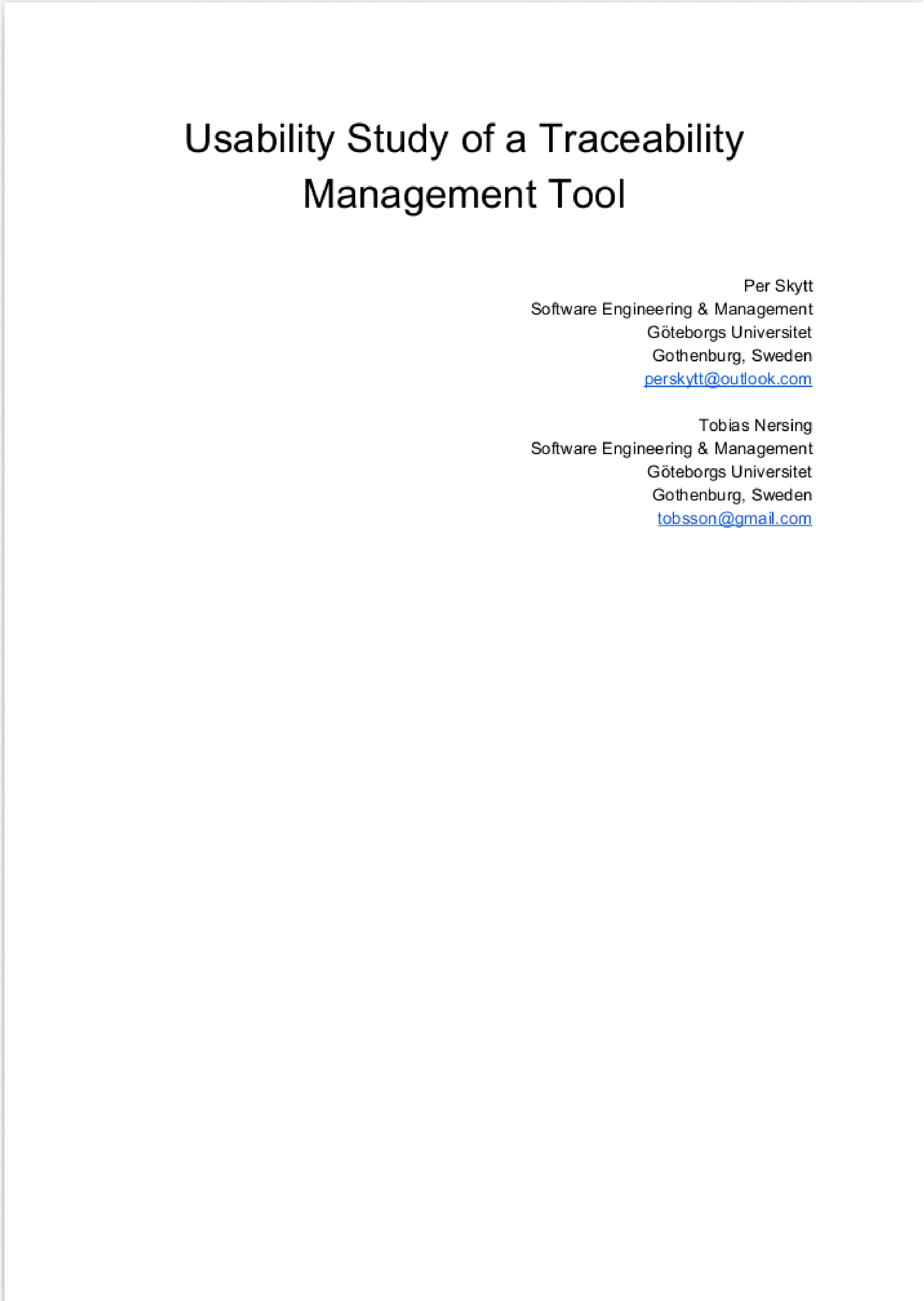
Attribute	Values		
Defect ID	UP12		
Frequency	3/6		
Trigger	Remove the trace link between ITOS feature and TemperatureAdapter Statechart.		
Context	User wants to remove a trace link that has already been created. User has added two artifacts to the Selection View and tries to find a way to remove the trace link there.		
Description	User tries to edit trace-links through the Selection View.		
Defect Removal Activity	User test		
Impact	**Severity**		Minor
	Task Efficiency(%/min)		39.61%
	Task Effectiveness	Completion Rate	66,67%
		Mean Time on Task(min)	01:41
Failure qualifier	Incongruent mental model		
Expected phase			
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP13		
Frequency	3/6		
Trigger	Open the Eclipse Problem view. Find the warning( ) concerning the deletedITOSTest java class and use the Eclipse “Quick Fix” function to remove all affectedtrace links.		
Context	User removes a trace link but doesn't save traceModel.xmi. After the quickfix in the problem view is applied, the popup appears.		
Description	Dialogue window asking user to discard changes made outside the editor (via quick-fix) confuses the user since the information given makes the consequences hard to understand (Eclipse pop-up)		
Defect Removal Activity	User test		
Impact	**Severity**		Moderate
	Task Efficiency(%/min)		52%
	Task Effectiveness	Completion Rate	100%
		Mean Time on Task(min)	01:55
Failure qualifier	Incongruent mental model		
Expected phase			
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP14		
Frequency	1/6		
Trigger	Delete a trace link that has already been created.		
Context	User is looking for where to delete a trace link. User doesn't realise that they have to look in the new folder created by Capra.		
Description	User failed to remove a trace in a previous task, they thought the trace was created (but the users missed to create the link in first place), user therefore have issues when asked to delete the trace from the tracemodel because the user can't find clear information of which links are created.		
Defect Removal Activity	User test		
Impact (average calculations of all relevant tasks)	**Severity**		Minor
	Task Efficiency(%/min)		39.61%
	Task Effectiveness	Completion Rate	66,67%
		Mean Time on Task(min)	01:41
Failure qualifier	Overlooked		
Expected phase	Presentation Design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

Attribute	Values		
Defect ID	UP15		
Frequency	1/6		
Trigger	Create a trace link between two different artifacts.		
Context	User wants to create a trace link between two artifacts. User has added two artifacts to the selection view and right clicks to create a trace link.		
Description	User is uncertain if items in Selection View need to be selected or not before issuing the command "Create Trace".		
Defect Removal Activity	User test		
Impact	Severity		Minor
	Task Efficiency(%/min)		65.71%
	Task Effectiveness	Completion Rate	99.66%
		Mean Time on Task(min)	01:31
Failure qualifier	Incongruent mental model		
Expected phase	Presenation design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			

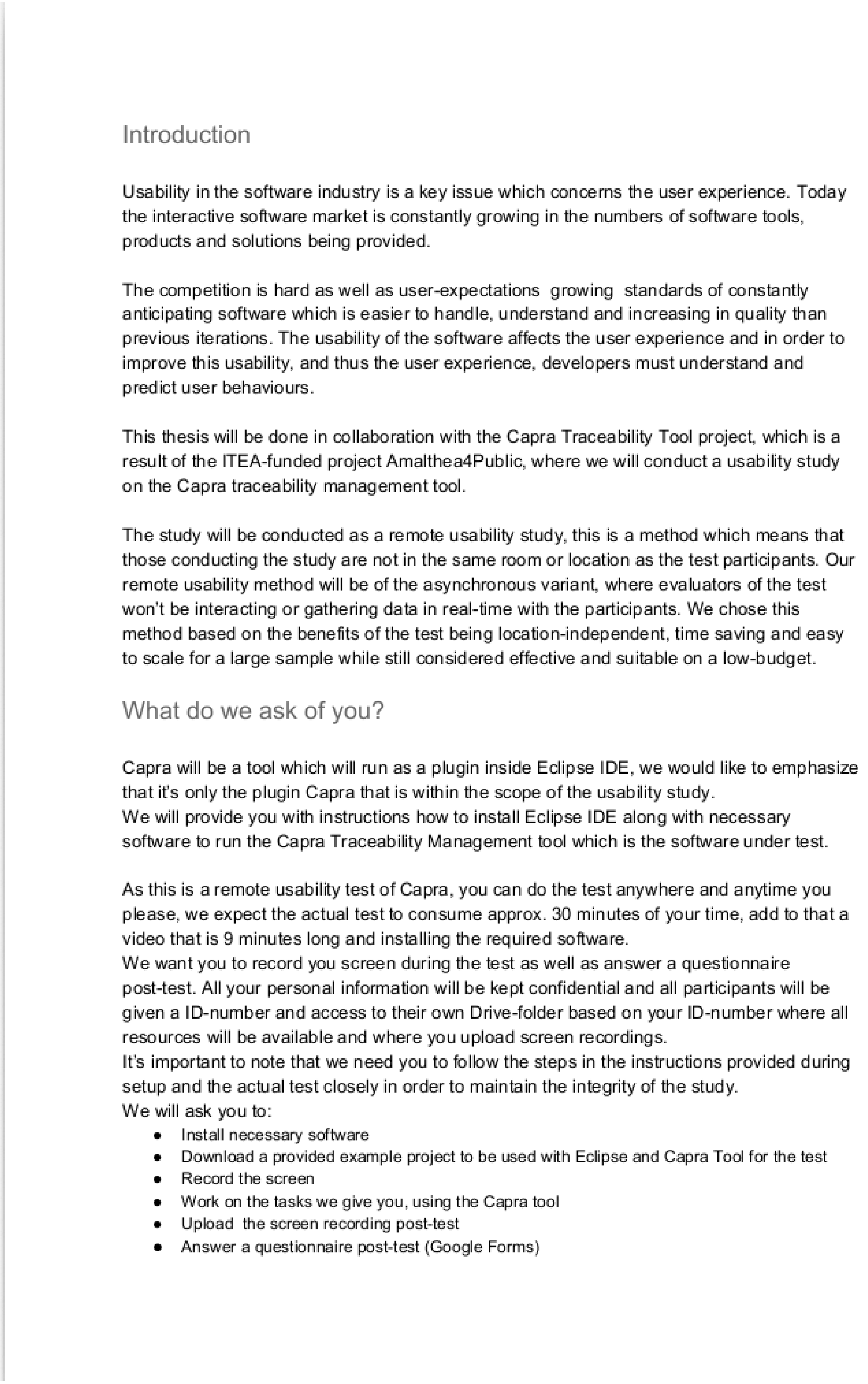
Attribute	Values		
Defect ID	16		
Frequency	6/6		
Trigger	View the Capra traceability matrix of HVAC_manager feature, TemperatureAdapterStatechart and ITOS feature to make sure ITOS feature isn't linked to the other two.		
Context	User wants to create a matrix from different artifact types. User opens different types of artifacts to try and find where all the artifacts can all be selected.		
Description	Problem with selecting multiple artifacts of different types to show in a matrix.		
Defect Removal Activity	User test		
Impact	Severity		Severe
	Task Efficiency(%/min)		25.35%
	Task Effectiveness	Completion Rate	91,67%
		Mean Time on Task(min)	03:37
Failure qualifier	Overlooked		
Expected phase	Presentation design		
Actual Phase			
Type of Fault Removed			
Cause:			
Error Prevention			



Usability Study of a Traceability Management Tool

Per Skytt
Software Engineering & Management
Göteborgs Universitet
Gothenburg, Sweden
perskytt@outlook.com

Tobias Nersing
Software Engineering & Management
Göteborgs Universitet
Gothenburg, Sweden
tobsson@gmail.com



Introduction

Usability in the software industry is a key issue which concerns the user experience. Today the interactive software market is constantly growing in the numbers of software tools, products and solutions being provided.

The competition is hard as well as user-expectations growing standards of constantly anticipating software which is easier to handle, understand and increasing in quality than previous iterations. The usability of the software affects the user experience and in order to improve this usability, and thus the user experience, developers must understand and predict user behaviours.

This thesis will be done in collaboration with the Capra Traceability Tool project, which is a result of the ITEA-funded project Amalthea4Public, where we will conduct a usability study on the Capra traceability management tool.

The study will be conducted as a remote usability study, this is a method which means that those conducting the study are not in the same room or location as the test participants. Our remote usability method will be of the asynchronous variant, where evaluators of the test won't be interacting or gathering data in real-time with the participants. We chose this method based on the benefits of the test being location-independent, time saving and easy to scale for a large sample while still considered effective and suitable on a low-budget.

What do we ask of you?

Capra will be a tool which will run as a plugin inside Eclipse IDE, we would like to emphasize that it's only the plugin Capra that is within the scope of the usability study.
We will provide you with instructions how to install Eclipse IDE along with necessary software to run the Capra Traceability Management tool which is the software under test.

As this is a remote usability test of Capra, you can do the test anywhere and anytime you please, we expect the actual test to consume approx. 30 minutes of your time, add to that a video that is 9 minutes long and installing the required software.
We want you to record you screen during the test as well as answer a questionnaire post-test. All your personal information will be kept confidential and all participants will be given a ID-number and access to their own Drive-folder based on your ID-number where all resources will be available and where you upload screen recordings.
It's important to note that we need you to follow the steps in the instructions provided during setup and the actual test closely in order to maintain the integrity of the study.
We will ask you to:

- Install necessary software
- Download a provided example project to be used with Eclipse and Capra Tool for the test
- Record the screen
- Work on the tasks we give you, using the Capra tool
- Upload the screen recording post-test
- Answer a questionnaire post-test (Google Forms)

SETUP & INSTALLATION

Before you actually start the test you need to follow the steps below to install necessary software and setup your computer. ***If you run into any problems contact us using the information at the end of this document***. It will take 10-15 minutes to get your system setup before you can start the test.

1. **INSTALLATION OF REQUIRED GRAPHVIZ SOFTWARE:**

LINUX AND WINDOWS USERS:

a. Open a browser
b. Go to: http://www.graphviz.org/Download..php
c. Select your OS in the menu to the far left under "Download"
d. Download the latest stable version of GraphViz suitable for your OS.
e. Install GraphViz using the downloaded msi-file(windows) OR .deb-file(Linux).

MAC USERS:

a. Open a terminal
b. You need to install Homebrew with this command:

```
/usr/bin/ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

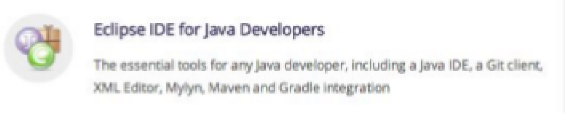
c. After Homebrew is installed copy/paste each of the following lines into a terminal, each line followed by pressing enter:

```
brew install libtool
brew link libtool
brew install graphviz
brew link --overwrite graphviz
```

2. **ECLIPSE NEON INSTALLATION**

a. Download Eclipse Installer from http://www.eclipse.org/downloads
b. Follow instructions here: https://eclipse.org/downloads/eclipse-packages/?show_instructions=TRUE

Select package:



Eclipse IDE for Java Developers

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Mylyn, Maven and Gradle integration

3. **INSTALLATION OF NEEDED PLUGINS TO USE TEST-PROJECT:**
    a. Open Eclipse IDE
    b. Click “Help” -> ”Install new software...”
    c. Copy a link from the list below and paste it in the “Work with”-textfield and press enter.
    d. Click “Select All” and then click the “Next”-button to start installing *(If nothing else is noted next to the link in the list below).*
    (Note: These steps can vary a lot in time for different users. If the installation appears to stop just give it time and it will complete.)

Do the above steps for each of the following links:

- **PlantUML**
  http://files.idi.ntnu.no/publish/plantuml/repository/

- **RMF**
  http://download.eclipse.org/rmf/updates/releases

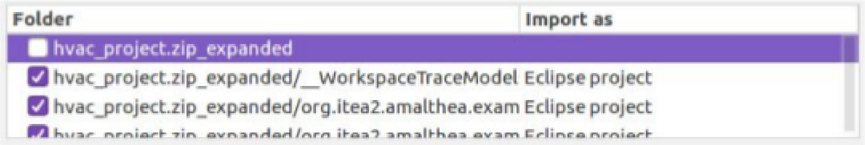
- **Capra**
  https://hudson.eclipse.org/capra/job/build-src-develop-nightly/lastSuccessfulBuild/artifact/org.eclipse.capra.releng.p2/target/repository/

- **Yakindu**
  http://updates.yakindu.org/sct/neon/releases/
  PS!!! DO NOT “Select All” packages, instead only select:
  Yakindu BASE
  Yakindu SCT

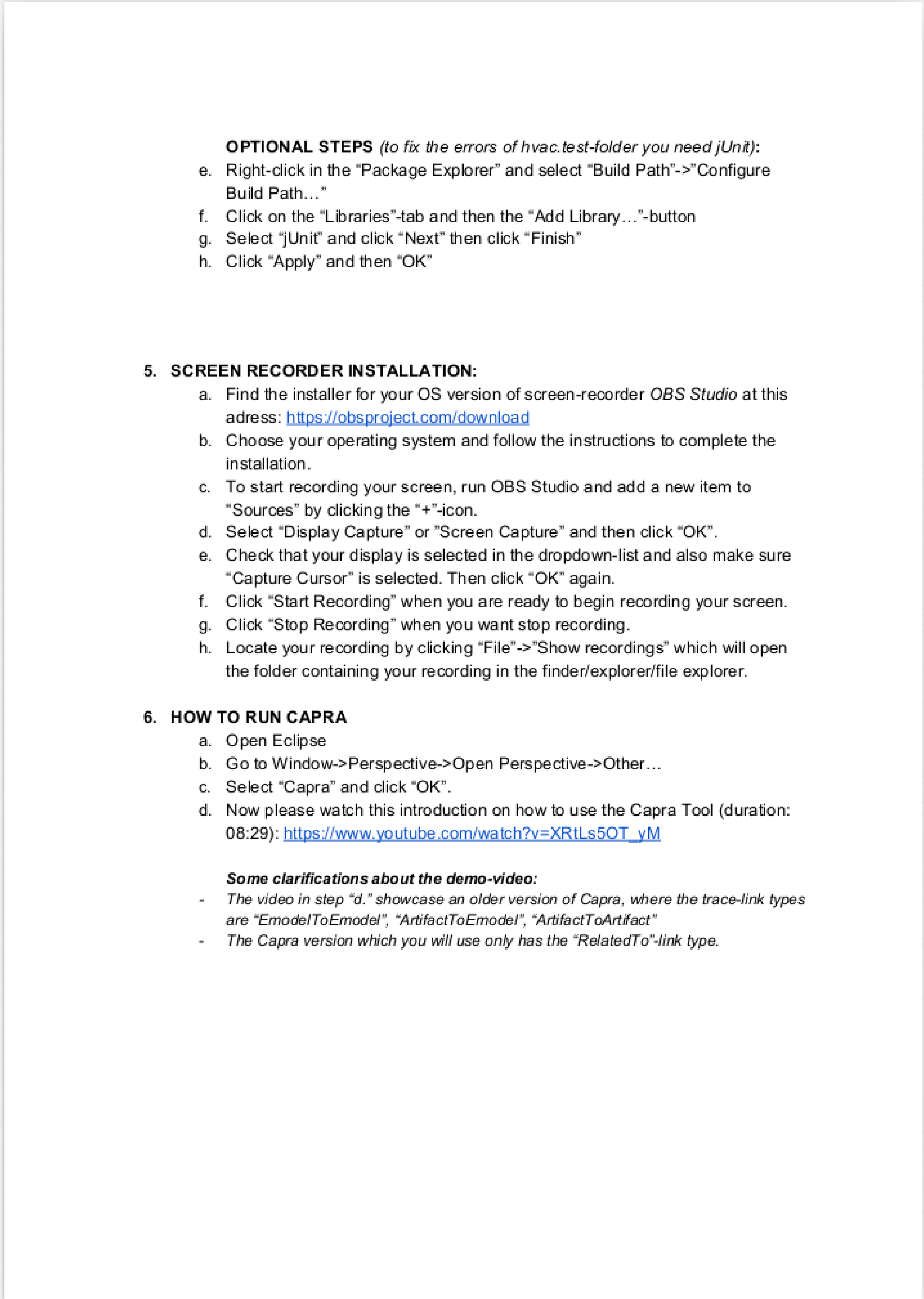
- **Franca**
  http://franca.github.io/franca/update_site/releases
  PS!!! DO NOT “Select All” packages, instead collapse “Franca” and only select:
  Franca Runtime
  Franca UI

- **Amalthea**
  http://platform.amalthea-project.org/update/
  PS!!! DO NOT “Select All” packages, instead only select:
  Amalthea Variability

4. **DOWNLOAD AND OPEN THE HVAC-PROJECT IN ECLIPSE:**
    a. Download the “hvac_project”-folder HERE.
    b. In Eclipse, close the welcome screen if open, then go to File->Import…
    c. Under “General” choose “Projects from Folder or Archive” and click “Next”
    d. Press the “Archive..”-button and select the downloaded hvac_project zip file, deselect “hvac_project.zip_expanded” then press “Finish”



Folder	Import as
□ hvac_project.zip_expanded	
☑ hvac_project.zip_expanded/__WorkspaceTraceModel	Eclipse project
☑ hvac_project.zip_expanded/org.itea2.amalthea.exam	Eclipse project
☑ hvac_project.zip_expanded/org.itea2.amalthea.exam	Eclipse project



OPTIONAL STEPS *(to fix the errors of hvac.test-folder you need jUnit)***:**

e. Right-click in the “Package Explorer” and select “Build Path”->”Configure Build Path…”
f. Click on the “Libraries”-tab and then the “Add Library…”-button
g. Select “jUnit” and click “Next” then click “Finish”
h. Click “Apply” and then “OK”

5. **SCREEN RECORDER INSTALLATION:**
   a. Find the installer for your OS version of screen-recorder *OBS Studio* at this adress: https://obsproject.com/download
   b. Choose your operating system and follow the instructions to complete the installation.
   c. To start recording your screen, run OBS Studio and add a new item to “Sources” by clicking the “+”-icon.
   d. Select “Display Capture” or ”Screen Capture” and then click “OK”.
   e. Check that your display is selected in the dropdown-list and also make sure “Capture Cursor” is selected. Then click “OK” again.
   f. Click “Start Recording” when you are ready to begin recording your screen.
   g. Click “Stop Recording” when you want stop recording.
   h. Locate your recording by clicking “File”->”Show recordings” which will open the folder containing your recording in the finder/explorer/file explorer.

6. **HOW TO RUN CAPRA**
   a. Open Eclipse
   b. Go to Window->Perspective->Open Perspective->Other…
   c. Select “Capra” and click “OK”.
   d. Now please watch this introduction on how to use the Capra Tool (duration: 08:29): https://www.youtube.com/watch?v=XRtLs5OT_yM

Some clarifications about the demo-video:

- *The video in step “d.” showcase an older version of Capra, where the trace-link types are “EmodelToEmodel”, “ArtifactToEmodel”, “ArtifactToArtifact”*
- *The Capra version which you will use only has the “RelatedTo”-link type.*

(Intentionally left blank)

Starting the test

When you've completed the setup and installation without problems you are ready to start the screen-recorder (if you use multiple displays please record both OR just interact on the single display you are recording during the test).

- Open Eclipse and use the Capra Perspective
- Make sure you have the HVAC_Project open in Eclipse
- Make sure have seen the demo-video
- Start recording your screen
- Perform the tasks below:

1. Create a trace link between the Requirement 4 and the ITOS feature.
2. Create a trace link between ITOS feature and TemperatureAdapter Statechart.
3. Create a trace link between TemperatureAdapter Statechart and the ITOSTest java class.
4. View the trace links of the ITOS feature through PlantUML diagram.
5. Use transitivity-function to see the whole trace of connections in relation to the ITOS feature.
6. Remove the trace link between ITOS feature and TemperatureAdapter Statechart.
7. Delete the ITOSTest java class.
8. Open the Eclipse Problem view. Find the warning(⚠) concerning the deleted ITOSTest java class and use the Eclipse "Quick Fix" function to remove all affected trace links.
   - To add the Problems view to the current perspective go to: Window->Show View->Other… and in "General"-folder select "Problems")
   - To access "Quick Fix"-option click on the warnings
9. Create a trace link between Requirement 4 and HVAC_manager feature.
10. Create a trace link between HVAC_manager feature and TemperatureAdapter Statechart.
11. View the trace links of HVAC_manager feature through PlantUML diagram.
12. View the Capra traceability matrix of HVAC_manager feature, TemperatureAdapter Statechart and ITOS feature to make sure ITOS feature isn't linked to the other two.

When you have finished the tasks:

- Stop the recording of your screen.
- Go to this Google Form and fill out the questionnaire (along with you Participant-ID#)
- Upload your screen-recording to your Participant Drive-folder