

System porting to mobile devices at the example of the SEE project

Master Thesis

Roman Gressler

Matriculation number: 3217822

June 7, 2022



Faculty 3 — Mathematics and Computer Science
Computer Science

1. Supervisor: Prof. Dr. Rainer Koschke
2. Supervisor: Prof. Dr. Zwetachter

ABSTRACT

TODO: Hier das Abstract der Arbeit. Kann deaktiviert werden.

ERKLÄRUNG

Ich versichere, diese Arbeit — sofern dies nicht explizit anders gekennzeichnet wurde — ohne fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen sind, sind als solche kenntlich gemacht.

Bremen, den June 7, 2022

Roman Gressler

ACKNOWLEDGEMENT

I want to thank me and only me for putting all this hard work in.

CONTENTS

1	Introduction	1
1.1	Motivation	1
1.2	Research Question	1
2	Concept	3
2.1	Interface	3
2.2	Interaction	7
2.3	Requirements	8
3	Implementation	11
4	Evaluation	13
4.1	SEE Desktop	13
4.2	Aim and hypothesis	14
4.3	Experiment set up	16
4.4	Realization	19
4.4.1	Survey tool	19
4.4.2	Questionnaires	20
4.4.3	Pilot study	22
4.4.4	Final experiment set up	22
4.4.5	Execution	25
4.4.6	Results	25
5	Conclusion	27
5.1	Outlook	27
A	Glossary	29
B	Acronyms	31
C	List of Figures	33
D	List of Tables	35

INTRODUCTION

1.1 MOTIVATION

1.2 RESEARCH QUESTION

The central research question of this thesis is: Are Android smartphones suitable of working with *Software Engineering Experience* (SEE)?

To answer this question this thesis will discuss a general concept to implement SEE on mobile devices in chapter 2. Afterwards the specific details of implementation will be introduced in chapter 3. The research question will then be answered in chapter 4 with an evaluation where the implemented mobile version of SEE will be compared with the desktop version. Last but not least this thesis will discuss further research and give a conclusion in chapter 5.

SEE: An interactive software visualization that uses the code-city metaphor and enables collaborative multiplayer interactions via multiple platforms like desktop, virtual reality and soon Android devices.

2

CONCEPT

In this section a concept of a mobile SEE version will be presented. Therefore, a prototype will be created to point out the features that a mobile version of SEE requires.

Prototypes are a common way to express the needs of a system. It is a low-cost way of planning an implementation, that can highlight challenges regarding constraints of a system early on.

Even though a prototype will never be able to show every aspect and need of a complex system, it should still help to answering questions like: How should the system feel? How should it be implemented, and what are the key features? [Houde and Hill \(1997\)](#)

SEE is meant to be used by multiple platforms such as desktop devices, mobile devices and virtual reality devices. Each device has different interaction constraints. While a desktop user will control the player with mouse and keyboard a mobile user will interact with virtual joysticks on a touchscreen. Selecting nodes of a [Code-City](#) will be done by clicking it with a mouse on desktop devices, while a mobile device will require a touch input.

[Code-City:](#) In the Code-City metaphor, software components are represented by buildings in a city, and the properties of these buildings can express different metrics of the software. For example the height of a "Node" could represent the lines of code in that class.

2.1 INTERFACE

In the following a paper prototype will be presented that marks out a concept for the mobile interface. Since the field of mobile development is quite young there few guidelines regarding the design of mobile device interfaces. A guideline that is widely accepted is problematic to find. [Renaud and Van Biljon \(2017\)](#), [Punchoojit and Hongwarittorn \(2017\)](#)

Major differences to desktop environments are the screen size, forms of input and input feedback. To assure as much space is used for the actual interaction of the app the menu should just take as much space as needed. As a study has found out, a size of at least 8*8 mm is needed to reduce error rates selecting the right button. [Conradi et al. \(2015\)](#) [Parhi et al. \(2006\)](#) TODO WEITER AUSFÜHREN SHORTCUTS WIE STRG Z NICHT MÖGLICH [Adipat and Zhang \(2005\)](#)

Moving the player will be handled with virtual joysticks as seen in figure 2.1. The left joystick will move the player through the virtual room and the right will move the camera angle or in other word the direction the player looks at. The joysticks are placed in the left and right corner and should just take as much space as needed to be handled

comfortably. This way the player is able to navigate through the virtual room with his/her thumbs while still having enough space to work on the Code-City.



Figure 2.1: Joysticks for moving in SEE

The menu on the top left side seen in figure 2.2 will be called “quickbar” further on. The quickbar can be minimized to save screen space when not needed. The quickbar is designed to offer more general functions that are needed in various situations. Because there are no shortcuts on mobile devices each function has to have a button to be activated.

The functions are redo and undo which will do an action undone again or revert an action. Then there is a camera lock that will lock the players perspective to a certain Code-City so that the player can only move around the selected city and move closer or further away from it. The next function is to rerotate a Code-City. That means the Code-City that was last rotated will be set back to its initial state of rotation. Last but not least there will be a button for recentering the city, which will work quite similar to the rerotate button and center the last moved Code-City. The button on the right can be used to collapse or expand the quickbar.

On the top right side another menu will be placed that contains different interaction modes. By clicking a button an interaction mode will be selected and moved to the top right corner. Also, the menu will be collapsed and only the buttons regarding the selected interaction mode shall be shown. By clicking the button on the top right again the menu shall expand and the other interaction modes shall be selectable. The other buttons shall be kept in the same order to reduce confusion of the user.

The first interaction mode, seen in figure 2.3, is for selecting nodes. Nodes can be selected by being touched and deselected by being touched again. There can be multiple nodes selected at once. The hole selection can be deselected by clicking the deselect button next to the

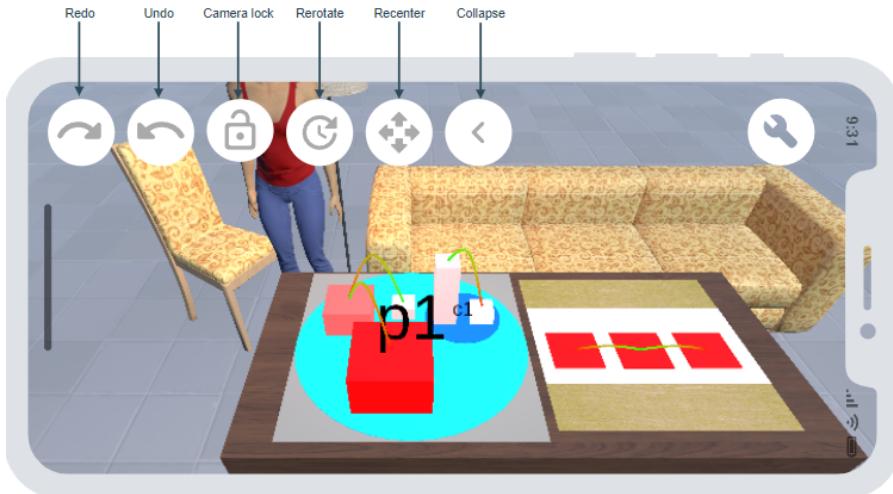


Figure 2.2: Quickbar for various interactions in SEE

select interaction mode button. Selected nodes shall be highlighted with a different node color and also display their name.

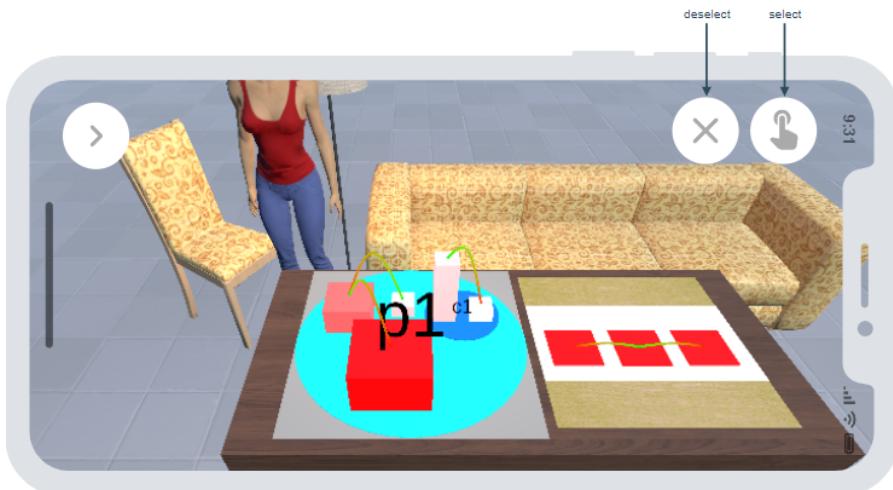


Figure 2.3: Selection mode in SEE

The second interaction mode, seen in figure 2.4, is for deleting node. It does not need additional buttons. Node will be deleted by being touched.- Unlike in the desktop version there will not be a group deletion interaction because it would require an additional menu panel. The added functionality would be minimal and selecting a group of nodes, confirming and finally deleting would require a handful more steps and would therefore most likely not be used.

The following interaction mode, seen in figure 2.5, is dedicated to the nodes and edges of a Code-City. Starting on with the “add node” button on the right. When activated the user can create new node by clicking on a certain spot on the Code-City plane. The following button on the left is for adding edges. By selecting two nodes a new edge will be created between them. Then, the button one further on the

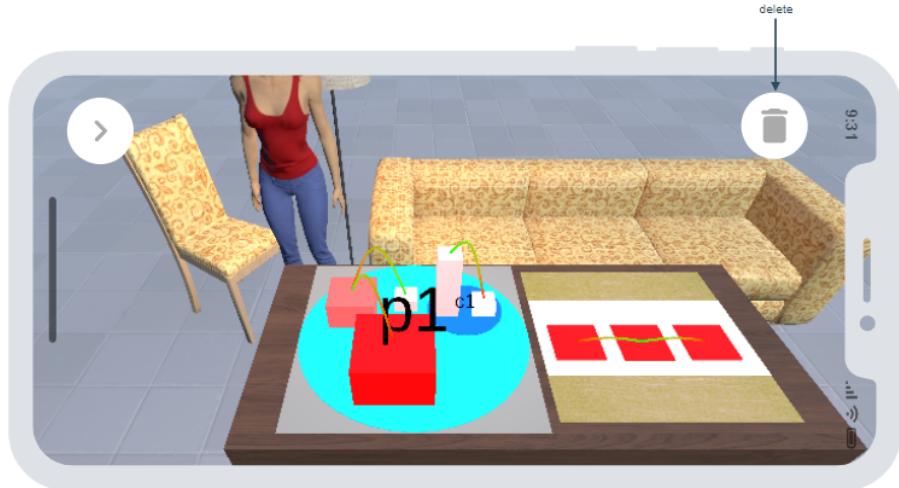


Figure 2.4: Delete mode in SEE

right is for editing nodes. By touching a node a window will pop up that allows the user to edit the node by changing its name and its type. Last but not least the button on the left-hand side will be used to scale nodes. That means the node height and width can be adjusted by first selecting it via touch and then hold a corner and slide it further away from the node center to increase the size or slide it towards the center to decrease the size of the node. Each button of the node interactions will be marked green after being pressed to indicate that it is active.

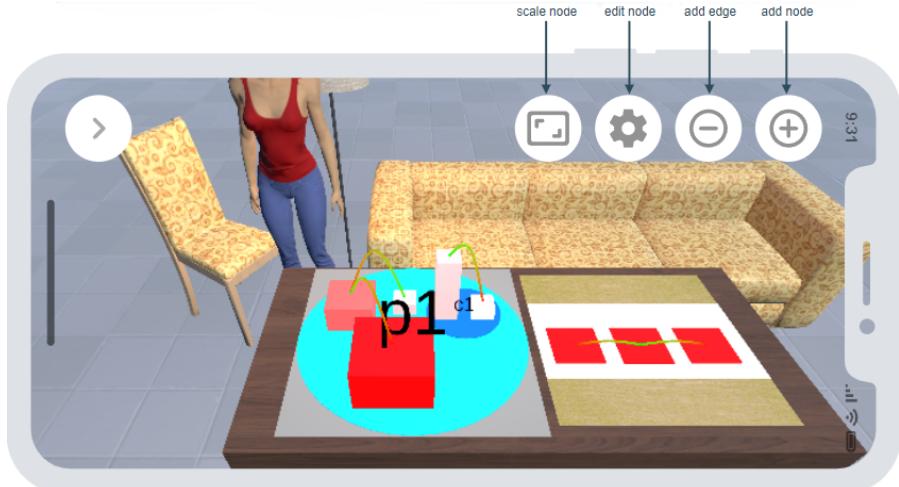


Figure 2.5: Node interactions in SEE

Then there will be a button for rotation interactions that can be seen in figure 2.6. Starting with the first activatable button that lets the user rotate the hole Code-City by touching any point on it and then sliding away from that point. Similar to that there will be a button that lets the user rotate just a single node on the Code-City. In addition to that there will be a button that activates the so-called “locked-rotation” mode. While in “locked-rotation” mode the rotation of a node or Code-City

will be done in eight predefined steps to a full rotation. Each step will have the same 45° range. The last button of this group will be for changing the center of the rotations. There are two options: the first option is a center of rotation in the middle of the Code-City and the second is in the middle of a node selection made with the interactions seen in figure 2.3. The second option can be activated by pressing the last button.



Figure 2.6: Rotation mode in SEE

The last interaction group, seen in figure 2.7, is for moving the Code-City or a single node. The move interactions are quite similar to the rotation interactions. There will be a button to move a hole Code-City as well as a button to move only single nodes. In addition to that there will be a button that restricts the movement of the Code-City or node to a predefined direction. The directions will be again in 45° angles and objects can be moved on a straight line on that angle. Moving a node or a Code-City can be achieved by touching and holding it and then moving it to the desired position.

2.2 INTERACTION

Smartphones are quite limited in space and there are few input possibilities. Unlike a desktop computer there is no mouse and there is no physical keyboard. Smartphones use virtual keyboards but due to the restriction of screen space the keyboard is hidden most of the time. Which would make keyboard shortcuts uncomfortable because the user has to open the keyboard first. Therefore, smartphones need different ways of interaction such as touch gestures.

Zooming in to a Code-City happens by scrolling on a desktop environment. There is no option to scroll on mobile devices, but there are at least two popular alternatives. The first option would be to double tap on the Code-City to zoom in. The double tap would zoom in, in



Figure 2.7: Movement mode in SEE

predefined steps and after reaching a certain level of closeness it would trigger to zoom out again. In SEE zooming in, in predefined steps might not be precise enough because there could be a quite large Code-City or a rather small one. Finding predefined steps that would fit every situation is rather hard. Therefore, a second option by zooming in with a two finger gesture might be better. In this option the user uses two fingers and slides them towards each other to zoom in or slides the two fingers away from each other to zoom out. This way there are no predefined steps necessary and zooming interactions can be done precisely.

2.3 REQUIREMENTS

In the following a list of requirements will be given, which will specify in detail what the implementation of a mobile version has to take care of. The list will be referred to multiple times in the upcoming realization part in chapter 3. Requirements are essential for the planning phase as they give a good fundamental structure for the developer to rely on. [Robertson and Robertson \(2012\)](#); [Stevens and Pooley \(2005\)](#)

- [R1] The application shall run on Android devices
- [R2] The application shall be controlled via touchscreen
 - [R2.1] The player and camera shall be moved with virtual joysticks
 - [R2.2] Needed shortcuts of the desktop version shall be handled with buttons
 - [R2.3] Zooming shall be handled with a two finger gesture
- [R3] The user shall be able to select a node of a Code-City
 - [R3.1] After selecting the name of the node shall be shown

[R3.2] The user shall be able to deselect single nodes or a group of nodes

[R4] The user shall be able to delete nodes

[R5] The user shall be able to interact with nodes

[R5.1] The user shall be able to add nodes

[R5.2] The user shall be able to add edges

[R5.3] The user shall be able to edit nodes

[R5.4] The user shall be able to scale nodes

[R6] The user shall be able to rotate a Code-City

[R6.1] The user shall be able to rotate a Code-City in 45° steps

[R6.2] The user shall be able to rotate single objects

[R6.3] The user shall be able to rotate around a center of selected nodes

[R6.4] The user shall be able to undo the rotation

[R7] The user shall be able to move a Code-City

[R7.1] The user shall be able to move single object of a Code-City

[R7.2] The user shall be able to restore the Code-City initial position

[R7.3] The user shall be able to move a Code-City or single node in predefined directions

[R8] The user shall be able to undo and redo actions

[R9] The user shall be able to lock the camera to a selected Code-City

3

IMPLEMENTATION

...

4

EVALUATION

In the following chapter the mobile implication of SEE will be evaluated in a user study. Therefore, the mobile application will be compared with the desktop version.

This chapter will start with a description of the desktop version and its main differences in section 4.1. Continuing with a defined aim and the precise hypotheses for the user study in section 4.2. After sketching the first experiment set up in section 4.3 the actual experiment set up will be discussed in detail in section 4.4 including the used survey tool, questionnaires and the pilot study.

4.1 SEE DESKTOP

In this section the desktop version of SEE will be explained. In this evaluation the mobile version of SEE will be compared with the desktop version. Therefore, it is necessary to take a deeper look at the differences between those two versions. Especially at how the interactions differ and what impact it could have on the user experience.

One outstanding difference from the desktop version to the mobile version is the selection of the interaction modes. While in the mobile version the menu for the interaction modes is always visible in the desktop version by pressing space a menu screen opens as seen in figure 4.1. Alternatively interaction modes can be changed by pressing one of the “1-9” keys, which, however requires the user to memorize which number belongs to which mode.

Another difference is the type of user input. The desktop version uses mouse hovering to display the name of a hovered *Node* or *Plane*. This is a faster method than touching the object first in the mobile version. In addition to that in the mobile version the object also has to be deselected otherwise there will be a lot of Node and Plane names displayed, and it will soon get quite messy. Also, the precision of object selection differs because touch input can never be as precise as selecting with a mouse cursor. This could force the mobile user to zoom further in because with a touch input it will not be possible to select small objects like it might be with a cursor. Which, of course, would require more time.

One more key difference is the available keyboard for desktop users. It allows using *Shortcuts*, which makes some menu items unnecessary but also requires the user to memorize those Shortcuts. The desktop

Node: A point in a diagram where lines intersect. In “SEE” it usually displays a software class

Plane: An area that bundles “nodes”. Could for example represent a namespace.

Shortcut: A combination of key that will call an action like for example “ctrl” + “c” for coping a text.

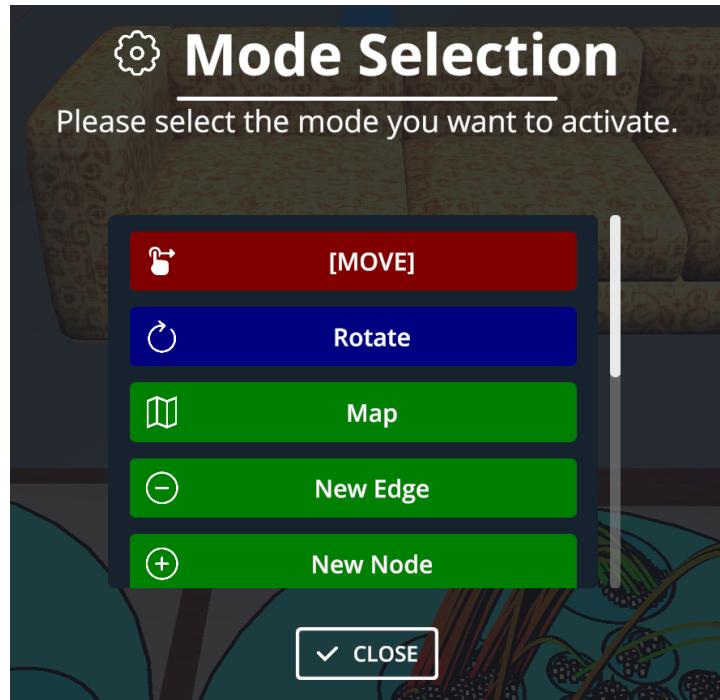


Figure 4.1: The desktop menu for selecting interaction modes.

version for example uses the “R” key in the move and rotation mode to recenter or rerotate a Code-City. In the mobile version on the other side the user will find a button for both actions. With the right amount of training both actions should probably equal in the amount of time they need but the mobile version sacrifices screen space for those buttons. If however the user has to type more text like in renaming objects, the common desktop keyboard should come in handy as a study from [Kim et al. \(2014\)](#) shows that even at a same keyboard size, a virtual one will lack in productivity.

4.2 AIM AND HYPOTHESIS

The aim of this user study is to answer the research question discussed in section 1.2. In order of answering the research question the finished prototype of the mobile extension shall be evaluated. Therefore, the system shall be compared on Android smartphones as well as desktop computers. Comparing these two use cases shall give insight on how much impact the constraints of mobile devices have on the usability and overall user experience. To measure the difference between the desktop and the mobile version the following hypotheses will be used. The two aspects performance and usability will be measured in the following study and each aspect will have a null hypothesis and an alternative hypothesis.

a) **Performance:** The time required for a task in SEE desktop will be called t_D and for mobile t_M .

- *Null Hypothesis H_{a0} :* The time required in SEE mobile is higher or the same as the time required in SEE desktop: $t_M \geq t_D$
- *Alternative Hypothesis H_{a1} :* The time required in SEE mobile is lower as the time required in SEE desktop: $t_M < t_D$

b) **Usability:** Two aspects are measured for *Usability*. First the *After-Scenario Questionnaire (ASQ)*-Score as a *Post-Task* result and second the *System Usability Scale (SUS)*-Score as a *Post-Study* result.

i) **ASQ:** Once again the aspect has to be split into three child aspects, because the three questions of the ASQ are independent:

- 1) The ASQ-Score for *effort* for SEE desktop is called A_{eD} and for SEE mobile is called A_{eM}

- *Null Hypothesis H_{b0} :* The ASQ-Score for *effort* is higher or even for SEE desktop than on SEE mobile: $A_{eD} \geq A_{eM}$
- *Alternative Hypothesis H_{b1} :* The ASQ-Score for *effort* is lower for SEE desktop than on SEE mobile: $A_{eD} < A_{eM}$

- 2) The ASQ-Score for *complexity* for SEE desktop is called A_{cD} and for SEE mobile is called A_{cM}

- *Null Hypothesis H_{c0} :* The ASQ-Score for *complexity* is higher or even for SEE desktop than on SEE mobile: $A_{cD} \geq A_{cM}$
- *Alternative Hypothesis H_{c1} :* The ASQ-Score for *complexity* is lower for SEE desktop than on SEE mobile: $A_{cD} < A_{cM}$

- 3) The ASQ-Score for *information* for SEE desktop is called A_{iD} and for SEE mobile is called A_{iM}

- *Null Hypothesis H_{d0} :* The ASQ-Score for *information* is higher or even for SEE desktop than on SEE mobile: $A_{iD} \geq A_{iM}$
- *Alternative Hypothesis H_{d1} :* The ASQ-Score for *information* is lower for SEE desktop than on SEE mobile: $A_{iD} < A_{iM}$

ii) **SUS:** The SUS-Score is called S_D for SEE desktop and S_M for SEE mobile.

- *Null Hypothesis H_{e0} :* The SUS-Score is higher or even for SEE desktop than for SEE mobile: $S_D \geq S_M$
- *Alternative Hypothesis H_{e1} :* The SUS-Score is lower for SEE desktop than for SEE mobile: $S_D < S_M$

Usability: A term that describes how well a (software) system can be used.

ASQ: A post-task questionnaire consisting of three questions that is used to access how difficult a user perceived a task (See "Post-Task").

Post-Task: A questionnaire that is taken after every task of an experiment.

SUS: The System Usability Scale consists of ten questions that measure "Usability"

Post-Study: A questionnaire that is taken after every block of an experiment.

The experiment will be participated different groups:

1. **SEE-developer:** They are already experienced with SEE-desktop. They are also experienced with software development and with first person games because they tried at least SEE itself, which counts as first person game experience.
2. **Non-SEE-developer:** This group has to be divided into four subgroups as follows:
 - **Software development and third person game experience:** They are more likely to understand the Code-City metaphor and are also more likely to be comfortable with the controls in SEE.
 - **Software development experience:** They are more likely to understand the Code-City metaphor and therefore might be able to find nodes faster.
 - **First person game experience:** They are also more likely to be comfortable with the controls in SEE.
 - **No experience:** They do not benefit from experience and therefore have to learn the most to interact in SEE

After the experiment the two versions of SEE can be compared as well as all above listed groups. This should give a detailed answer to the research question of this thesis.

4.3 EXPERIMENT SET UP

The system shall be tested in two groups each starting with a different device. Each group does the test on both devices, but one group will start with the mobile application and the other one with the desktop application. The participants will be assigned random to the groups. The testers will have various tasks to test the usability of the two applications. Afterwards the users will get a survey in English to document their impressions.

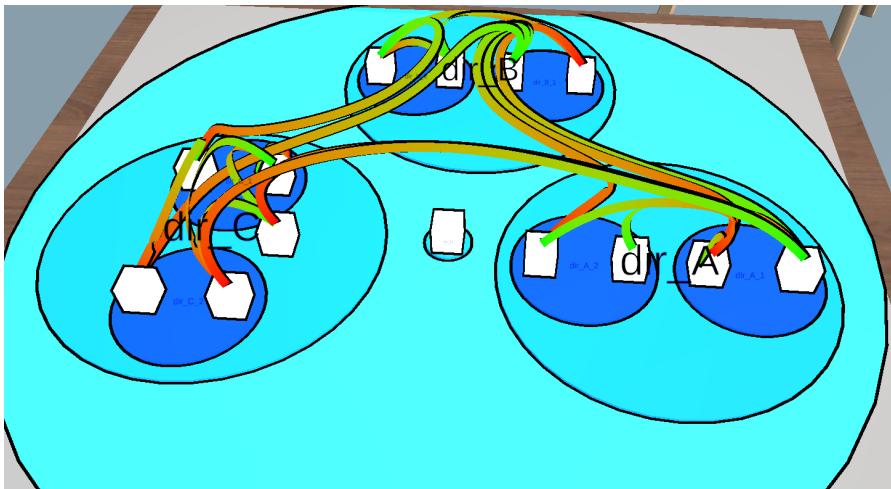


Figure 4.2: The first Code-City for the user study

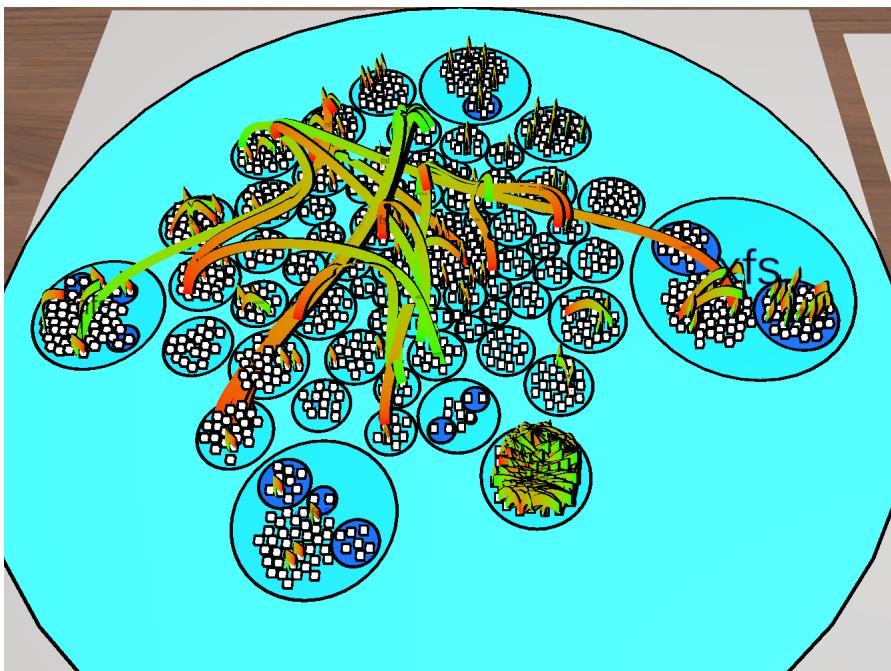


Figure 4.3: The second Code-City for the user study

In this survey the subjects will be asked various demographic questions as well as what Android device and version they will be using. In addition to that the subjects will be asked if they are experienced with SEE and if they are experienced with software development. Before the subjects will be asked to solve various tasks they will be asked to watch a short tutorial video on each application. After the video they will get a training task where every subject can get used to the system and ask questions if they have trouble solving the training task. The overseer will also make sure that every essential action will be practiced such as zooming and moving the Code-City. Figure 4.2 shows a small arranged

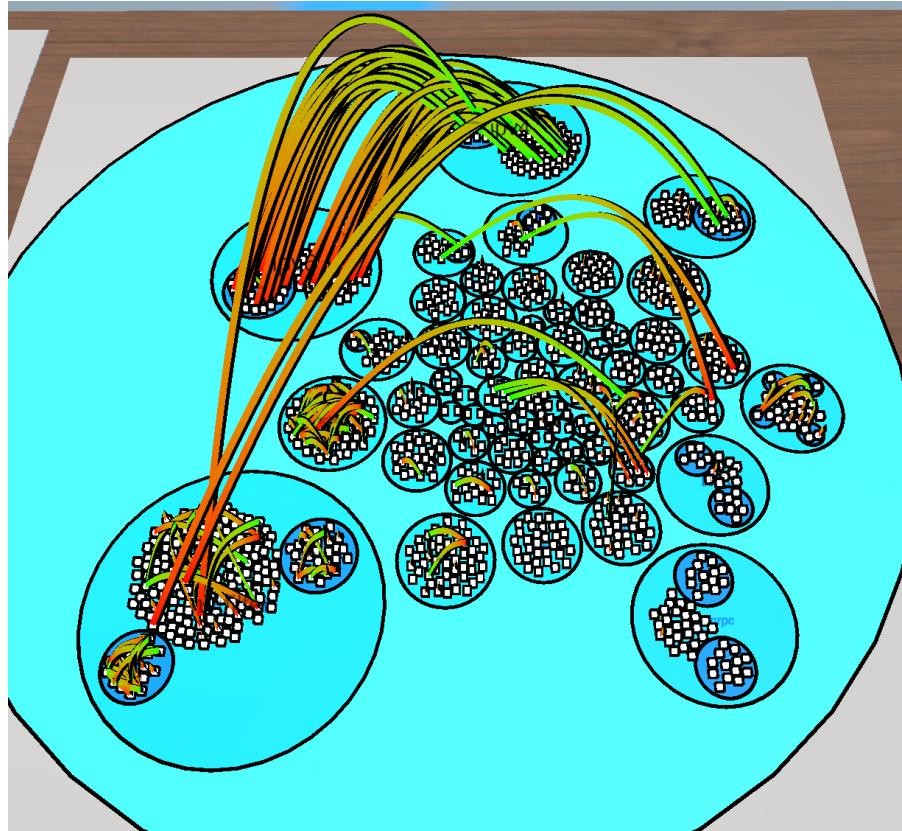


Figure 4.4: The third Code-City for the user study

Code-City that shall be used for the training tasks. The structure of the training Code-City is generic and follows a simple pattern. That shall ensure that the user can focus on the training and that the user does not get overwhelmed.

Following the first questions and the training, the subjects can start with the main tasks. For each application there will be two tasks and after each task the subjects will be handed a Post-Task questionnaire. Last but not least there will another questionnaire that aims to scale the Usability of the two applications. For the Post-Task questions the ASQ will be used and for the Usability questions SUS will be used. Both questionnaires will be discussed later on in section 4.4.2. For each main task the overseer will also take the completion time of every main task. The first and second task on the first device will be performed on the Code-City that can be seen in figure 4.3 and the third and forth task on device two will be performed on the Code-City that can be seen on figure 4.4. These examples are much larger than the training Code-City and represent real life code. The second Code-City shows the file system of Linux and the third one shows the network component of Linux. That way the tasks might reflect better on real world uses for SEE.

To not exhaust the testers too much the experiment shall not take longer than one hour. This also ensures that there is no to little variance

due to exhaustion. Each participant might have a different concentration span, but this shall not be the focus of this experiment.

4.4 REALIZATION

The following sections will cover the realization of the previously planned study. The choice of the used survey tool and questionnaires will be explained in section 4.4.1 and section 4.4.2. Afterwards a pilot study will be executed to test the study and possibly find missing aspects in section 4.4.3. Finally, the final experiment set up will be discussed in section 4.4.4.

4.4.1 Survey tool

As a survey tool Google Forms¹ will be used. The survey tool has to fulfill the following requirements:

- The study will be online because an overseer has to attend every experiment, and therefore it comes in handy to be flexible in terms of location. For this reason the survey shall be fully in a browser.
- The survey tool should be free to use.
- Subjects shall be anonymous.
- The results shall be exportable in a data format like *Comma-Separated Values (CSV)*
- Subjects should have the option to presave their answers. In addition to that answers should not be lost on reload.
- There should be an option to embed the introduction videos in the survey.

Google Forms fulfills all these requirements and will therefore be used. The final form can be seen in figure 4.5 which shows the intro of the survey and in figure 4.6 which shows the embedded intro video for SEE mobile.

CSV: A file format used for example to store table data. Each line represents a data record and each value is separated with a comma.

¹ <https://www.google.com/forms/about/> (last visit: 05.06.2022)

See Desktop - See Mobile

Thank you for participating in this study. The following questions will give insight on how well the different versions of SEE perform. The survey will take about an hour and the taken data will be kept anonymous.

SEE is project for code visualization that aims offer a collaborative space where developer can come together and have a look at various software projects. Therefore graphs are used that resemble cities. So called Code-Cities.

[In Google anmelden, um den Fortschritt zu speichern. Weitere Informationen](#)

[Weiter](#) [Alle Eingaben löschen](#)

Geben Sie niemals Passwörter über Google Formulare weiter.

Dieser Inhalt wurde nicht von Google erstellt und wird von Google auch nicht unterstützt. [Missbrauch melden](#) - [Nutzungsbedingungen](#) - [Datenschutzerklärung](#)

Google Formulare

Figure 4.5: The intro of the survey

See Mobile - See Desktop

[In Google anmelden, um den Fortschritt zu speichern. Weitere Informationen](#)

See Mobile

Please watch this short instruction of the app before you continue



[Zurück](#) [Weiter](#) [Alle Eingaben löschen](#)

Figure 4.6: The introduction video of the survey

4.4.2 Questionnaires

There will be three questionnaires used for the study that will be discussed in detail in the following. The study will start with a demographic questionnaire that covers general information about the

subject. After every task there will be ASQ and after every block of tasks for each of the two covered devices there will be a SUS questionnaire.

DEMOGRAPHIC QUESTIONNAIRE

The subjects shall start the survey with a demographic questionnaire. In that section they will be asked for their age, gender, the highest degree, experience with SEE, experience with first person video games, their Android device name, their Android version and their experience with software development. These specifications will be used to form different groups to see if there is any impact on the result of the following measurements. [Mclellan et al. \(2011\)](#) has shown that the user experience can have a significant impact on the SUS-score. It is therefore important to view the measured results in context of the paired demographic data.

The mobile version was only tested on a single Android device. It is likely that the performance of the application varies on different Android versions or devices.

POST-TASK QUESTIONNAIRE

The Post-Task questionnaire will supplement the Post-Study questionnaire on a micro level. The main focus here will be on single tasks, which allows to have a look at different aspects like effort, complexity and information provided by the system.

As a Post-Task questionnaire the ASQ will be used. The ASQ was first introduced in 1991 by [Lewis \(1991\)](#). It is designed for task based surveys and contains three questions. The ASQ will be used because it brings the following advantages:

- The questionnaire has been used many times over the years and has proven its validly ([Hajesmael-Gohari et al. \(2022\)](#); [Lewis \(1991\)](#); [Lewis \(1995\)](#)).
- With its three questions it is short and does not exhaust the subjects. This is especially important because it will be required to finish this questionnaire a total of four times.
- It fits well for this study because it is a questionnaire designed for task based evaluations.

The ASQ consists of three questions that scale from one to seven where one means “strongly disagree” and seven means “strongly agree”.

POST-STUDY QUESTIONNAIRE

The Post-Study questionnaire is mainly to obtain as much information about the Usability of the two systems as possible. The questionnaire can be longer than the Post-Task but still should not be to long to keep the processing time of the survey at around an hour.

The SUS questionnaire was first published in 1986 by John Brooke ([Brooke \(1996\)](#)) and is therefore widely used and proven as citations in

more than 1200 publications up until 2013 show (Brooke (2013)). The SUS is used in this study for the following advantages:

- It consists of ten questions and has to be done two times. Twenty questions in total fit well into the planned one hour total time span of this experiment.
- It has been made publicly available and is free to use (Brooke (1996)).
- It is widely used and therefore already proven to give useful results (Brooke (2013); Lewis (2018); Grier et al. (2013)).
- According to Peres et al. (2013) and Bangor et al. (2008) it is suited to compare two systems.
- Besides Usability Lewis and Sauro (2009) finds that also learnability is measured, which will also be looked at later on in section 4.4.6.

The SUS questionnaire will contain as already mentioned ten questions. Each question will give a five-level rating from “strongly disagree” to “strongly agree”. The results will then be combined into a Usability-score from zero to 100 where a high value represents a good result and a low value a bad one.

4.4.3 Pilot study

In a first test the pilot study was executed with one tester. Afterwards the study was discussed and checked for errors. It stood out that the example Code-City of task one was too different to the one in the second task. Therefore, the Code-City of the first task was exchanged with a larger and better comparable one. Further on a Code-City with 1288 nodes (see figure 4.3) as well as one with 1464 nodes (see figure 4.4) will be used.

Also, the tasks were not comparable because they differed in the types of interactions they used. In one task the user was asked to rename a node and in the other one the user shall add four nodes. For renaming a node the user has to use a keyboard which does not make it comparable to just click and add nodes in the second task.

4.4.4 Final experiment set up

Demographic questions:

- Age
 - 0-15 years old
 - 16-30 years old

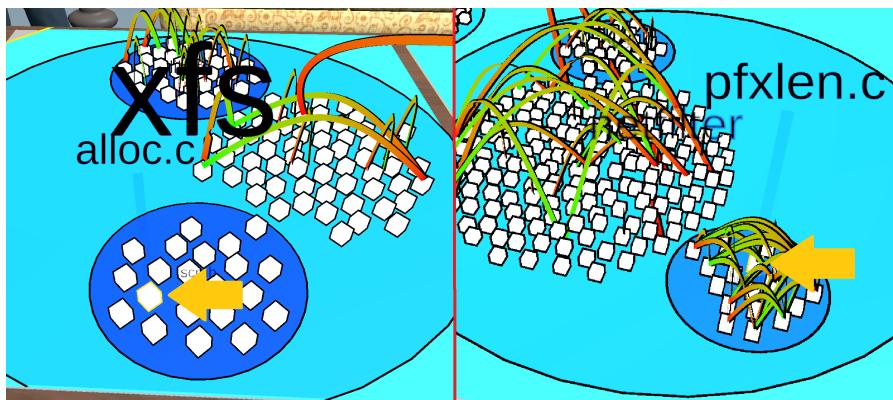


Figure 4.7: The two key nodes are marked with a yellow arrow

- 31-45 years old
- 46+ years old
- What gender do you identify as?
 - Male
 - Female
 - Other ...
 - Prefer not to say
- What is the highest degree or level of education you have completed?
 - Some High School (Hauptschule/Realschule...)
 - High School (Abitur)
 - Bachelor's Degree
 - Master's Degree
 - Ph.D. or higher
 - Prefer not to say
 - Other ...

Questions regarding used hardware and experience

- Are you already experienced with See?
- Do or did you play first person video games?
- Do or did you develop software?
- On which Android device will you attend?
- Which Android version are you using?*

Nr.	Task	Expected time
Training	Navigate through the planes "dir_root" >"dir_B" >"dir_B_2". On that plane select "b2_b.cpp" and rename it "b42".	1 - 5 mins
1	Detect the largest plane "xfs". On that plane find plane "scrub". Then find and delete node "alloc.c".	0.5 - 5 mins
2	Find the plane with one blue child plane ("btrfs"). On the blue child plane "tests" add four new nodes.	1 - 5 mins
Training	Navigate through the planes "dir_root" >"dir_C" >"dir_C_2". On that plane select "c2_b.cpp" and rename it "c42".	1 - 5 mins
3	Detect the largest plane "netfilter". On that plane find plane "ipset". Then find and delete node "pfxlen.c".	0.5 - 5 mins
4	On the plane with the most edges ("ipv6") find the smallest plane "ila" and connect all four nodes on it.	1 - 5 mins

Table 4.1: The tasks used for the experiment. The device will be switched after task 2.

Phase	Description		
Pre-Experiment	Demographic questionnaire		
	City	Group 1	Group 2
Training	Figure 4.2		
Task 1	Figure 4.3		
ASQ			
Task 2	Figure 4.3	Desktop	Mobile
ASQ			
SUS			
Training	Figure 4.2		
Task 3	Figure 4.4		
ASQ			
Task 4	Figure 4.4	Mobile	Desktop
ASQ			
SUS			

Table 4.2: Experimental procedure per subject. The procedure is swapped per group.

4.4.5 *Execution***4.4.6** *Results*

5

CONCLUSION

...

5.1 OUTLOOK

AR - [Santos et al. \(2016\)](#)

A

GLOSSARY

Code-City In the Code-City metaphor, software components are represented by buildings in a city, and the properties of these buildings can express different metrics of the software. For example the height of a “Node” could represent the lines of code in that class. [3–9](#), [14](#), [16–18](#), [21](#)

Node A point in a diagram where lines intersect. In “SEE” it usually displays a software class [3](#), [13](#), [16](#)

Plane An area that bundles “nodes”. Could for example represent a namespace. [13](#)

Post-Study A questionnaire that is taken after every block of an experiment. [15](#)

Post-Task A questionnaire that is taken after every task of an experiment. [15](#), [18](#), [29](#)

Shortcut A combination of key that will call an action like for example “ctrl” + “c” for coping a text. [13](#)

Usability A term that describes how well a (software) system can be used. [15](#), [18](#), [29](#)

B

ACRONYMS

ASQ A post-task questionnaire consisting of three questions that is used to access how difficult a user perceived a task (See “Post-Task”).
[15](#), [18](#), [21](#)

CSV A file format used for example to store table data. Each line represents a data record and each value is separated with a comma. [19](#)

SEE An interactive software visualization that uses the *code-city* metaphor and enables collaborative multiplayer interactions via multiple platforms like desktop, virtual reality and soon Android devices. [1](#), [3–8](#), [13](#), [15–19](#)

SUS The System Usability Scale consists of ten questions that measure “Usability” [15](#), [18](#), [21](#)

C

LIST OF FIGURES

Figure 2.1	Joysticks for moving in SEE	4
Figure 2.2	Quickbar for various interactions in SEE	5
Figure 2.3	Selection mode in SEE	5
Figure 2.4	Delete mode in SEE	6
Figure 2.5	Node interactions in SEE	6
Figure 2.6	Rotation mode in SEE	7
Figure 2.7	Movement mode in SEE	8
Figure 4.1	The desktop menu for selecting interaction modes.	14
Figure 4.2	The first Code-City for the user study	17
Figure 4.3	The second Code-City for the user study	17
Figure 4.4	The third Code-City for the user study	18
Figure 4.5	The intro of the survey	20
Figure 4.6	The introduction video of the survey	20
Figure 4.7	The two key nodes are marked with a yellow arrow	23

D

LIST OF TABLES

Table 4.1	The tasks used for the experiment. The device will be switched after task 2.	24
Table 4.2	Experimental procedure per subject. The procedure is swapped per group.	24

Regie: Kontrolliere am Ende, ob alle bibliographischen Angaben vollständig sind. Wird also die Zeitschrift oder Konferenz aufgeführt, in der ein Artikel veröffentlicht wurde? Sind überall die Seitenangabe aufgeführt? Bei Verweisen auf Web-Seiten, ist überall angegeben, wann der letzte Zugriff darauf erfolgte? Sind Umlaute und andere Sonderzeichen korrekt in LaTeX beschrieben worden?

BIBLIOGRAPHY

- Stephanie Houde and Charles Hill. What do prototypes prototype? In *Handbook of human-computer interaction*, pages 367–381. Elsevier, 1997.
- Karen Renaud and Judy Van Biljon. Demarcating mobile phone interface design guidelines to expedite selection. *South African Computer Journal*, 29(3):127–144, 2017.
- Lumpapun Punchoojit and Nuttanont Hongwarittorrn. Usability studies on mobile user interface design patterns: a systematic literature review. *Advances in Human-Computer Interaction*, 2017, 2017.
- Jessica Conradi, Olivia Busch, and Thomas Alexander. Optimal touch button size for the use of mobile devices while walking. *Procedia Manufacturing*, 3:387–394, 2015.
- Pekka Parhi, Amy K Karlson, and Benjamin B Bederson. Target size study for one-handed thumb use on small touchscreen devices. In *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, pages 203–210, 2006.
- Boonlit Adipat and Dongsong Zhang. Interface design for mobile applications. *AMCIS 2005 proceedings*, page 494, 2005.
- Suzanne Robertson and James Robertson. *Mastering the Requirements Process: Getting Requirements Right*. Addison-Wesley Professional, 2012. ISBN 978-0-13-294285-0.
- P. Stevens and R. Pooley. *Software Engineering with Objects and Components*. Springer, 2005.
- Jeong Ho Kim, Lovenoor Aulck, Michael C Bartha, Christy A Harper, and Peter W Johnson. Differences in typing forces, muscle activity, comfort, and typing performance among virtual, notebook, and desktop keyboards. *Applied ergonomics*, 45(6):1406–1413, 2014.
- Sam Mclellan, Andrew Muddimer, and S. Peres. The effect of experience on system usability scale ratings. *Journal of Usability Studies*, 7, 11 2011.
- James R Lewis. Psychometric evaluation of an after-scenario questionnaire for computer usability studies: the asq. *ACM Sigchi Bulletin*, 23 (1):78–81, 1991.
- Sadrieh Hajesmael-Gohari, Firoozeh Khordastan, Farhad Fatehi, Hamidreza Samzadeh, and Kambiz Bahaadinbeigy. The most used

- questionnaires for evaluating satisfaction, usability, acceptance, and quality outcomes of mobile health. *BMC Medical Informatics and Decision Making*, 22(1):1–9, 2022.
- James R Lewis. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human-Computer Interaction*, 7(1):57–78, 1995.
- John Brooke. Sus: a “quick and dirty’usability. *Usability evaluation in industry*, 189(3), 1996.
- John Brooke. Sus: a retrospective. *Journal of Usability Studies*, 8:29–40, 01 2013.
- James R Lewis. The system usability scale: past, present, and future. *International Journal of Human–Computer Interaction*, 34(7):577–590, 2018.
- Rebecca A Grier, Aaron Bangor, Philip Kortum, and S Camille Peres. The system usability scale: Beyond standard usability testing. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 57, pages 187–191. SAGE Publications Sage CA: Los Angeles, CA, 2013.
- S. Camille Peres, Tri Pham, and Ronald Phillips. Validation of the system usability scale (sus): Sus in the wild. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 57(1):192–196, 2013. doi: 10.1177/1541931213571043. URL <https://doi.org/10.1177/1541931213571043>.
- Aaron Bangor, Philip T. Kortum, and James T. Miller. An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24(6):574–594, 2008. doi: 10.1080/10447310802205776. URL <https://doi.org/10.1080/10447310802205776>.
- James R. Lewis and Jeff Sauro. The factor structure of the system usability scale. In Masaaki Kurosu, editor, *Human Centered Design*, pages 94–103, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-02806-9.
- Carlos Santos, Brunelli Miranda, Tiago Araujo, Nikolas Carneiro, Anderson Marques, Marcelle Mota, Jefferson Morais, and Bianchi Meiguins. Guidelines for graphical user interface design in mobile augmented reality applications. In *International Conference on Virtual, Augmented and Mixed Reality*, pages 71–80. Springer, 2016.