7-21-2017

Preserving the Spatial Information of Accessible UML Class Diagrams for the Visually Impaired

Silva Hekmat Matti
sxm4161@rit.edu

ROCHESTER INSTITUTE OF TECHNOLOGY

MASTERS THESIS

Preserving the Spatial Information of Accessible UML Class Diagrams for the Visually Impaired

Author:
Silva Hekmat MATTI

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering

Department of Software Engineering
B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, NY

July 21, 2017

Preserving the Spatial Information of
Accessible UML Class Diagrams for the
Visually Impaired
by
Silva Hekmat MATTI

Committee Approval:

Yasmine El-Glaly, Ph.D. July 13, 2017

Thesis Committee Chair

Stephanie Ludi, Ph.D. July 13, 2017

Committee Member

Scott Hawker, Ph.D. July 13, 2017

Graduate Program Director

Rochester Institute of Technology

Abstract

B. Thomas Golisano College of Computing and Information Sciences

Department of Software Engineering

Master of Science in Software Engineering

Preserving the Spatial Information of Accessible UML Class Diagrams for the Visually Impaired

by Silva Hekmat MATTI

One of the aspects that still needs to be fully accessible to persons with Visual Impairments or Blindness is programming and software engineering as a profession. UML diagrams are still an area to be improved when it comes to accessibility: How to make these diagrams more available? What kind of representation is more proper and efficient? This exploratory research suggests a scheme to navigate UML class diagram, with focus on presenting spatial information with related alternative text. In this research we try to find whether the suggested methodology is helping programmers and students with visual impairment in identifying and reading class diagrams more efficiently, and if they can build a cognitive map of the diagram. For testing and improving the suggested navigation scheme, we built a prototype and designed a study. We found that the navigational scheme is helping to find different connections and relations easily, but reaching a specific point with a jump or search function is needed. Also the participant were able to build a cognitive map of the class components.

This research is targeting a very specific user demography which includes persons with visual impairment who want to pursue Software Engineering as a profession, or persons who would have some classes in programming.

Acknowledgements

Foremost, I would like to express my gratitude to my advisor Dr. Stephanie Ludi for her continues support and guidance during my Masters study and research. This work wouldn't have been done without your patience, motivation and enthusiasm.

I would also like to thank my thesis committee members: Dr. Yasmine El-Glaly and Dr. Scott Hawker who is also my academic advisor. Their encouragement and insightful comments where a huge guidance.

My sincere thanks also goes to Dr. Daniel Krutz for his mentorship and guidance during the last two years.

I'm also grateful to the Fulbright program and my advisors in AMIDEAST.

This work wouldn't have been done without all the support of family, friends and the RIT community…

Contents

List of Figures

List of Tables

Dedicated to
My beautiful H&H family Hekmat&Hamama, Sandra&Sevan,
Sally and Matti…

Chapter 1

Introduction

1.1 Background

As computer and software are becoming part of everyday life, the need for more software engineers and developers is growing day by day. Studying software engineering and Software engineering as a profession should be completely open and available to anyone who would want to enter this discipline.

The dependency on images and graphical content has increased gradually in the computer and digital world. What used to be textual representation are being transformed into any sort of graphical representation with spatial information. Such graphic content are made available to people with visual impairments through different technologies and applications in computers and phones such as screen readers and magnifiers with proper sonification. Images are also made accessible through providing proper alternative text to the image. This should be created when creating the graphical content, which is not provided most of the time or it's too general. Also many laws and acts (*Section508.gov*) have been legislated to ensure that people must have equal access to everything. And also specifications have been defined and published to help make web content more accessible (*WAI-ARIA*).

Non-textual information can be provided for people with visual impairments using verbal description or text. However this text might be not representative of the image information, depending on the person who is providing this information. Another way is tactile prints. This way would preserve the spatial information of the image, however it would be costly, and only a very narrow subset of images would be transferred to tactile image and only on demand; which means a very limited access to such information for people with visual impairment. Finding a way that complements the previous one, with less costs and higher availability would be helpful.

Such transformations from textual to graphical are made on different levels and different ways. These transformations are made to imagine and understand systems in a better way. UML Diagrams are an example of such graphical representations, which are now a huge part of teaching programming for students and understanding software systems. UML is a standard modeling language with a rich graphical notation, and comprehensive set of diagrams and elements. An aspect that needs to be made fully accessible for persons with visual impairment who want to pursue profession in the Software and computer industry, or students with visual impairment who would want to have a one programming class is UML. As can be seen this dependency on visual contents is not only affecting people with visual impairment and blind when using computers and new technology, but also those who want to study or work in the computer related fields of study.

When creating UML diagrams, they are mostly saved and shared in an image format. Generally research about UML diagrams can take two directions; the first direction is how programmers or students with visual impairment could create a UML diagram; the second path is how to make existing diagrams that they might need or want to 'read' available for them. This research is mainly concerned with the second one.

Until now, different surveys show there are no well-known and famous technologies among programmers or students with visual impairment that help them read and create UML diagrams (Albusays and Ludi, 2016). Most of the previous research done to make graphs accessible is concerned of how to create graphs or diagrams that are accessible, which means that they will not have the access to the entire set of information that is available to everyone else, but only to the specific graphs that are made accessible. Other concerns are what level of detail is to be given and what kind of representation is sufficient to make the diagram understandable by a programmer or programming student with visual impairment.

For UML diagrams it's not only about giving a meaningful textual description but also how the UML information will be analyzed, represented and communicated in a way that factual and important information will not be lost; keeping in mind that a programmer with visual impairment might not be working alone but with a team of sighted people. Presenting the spatial positioning of the nodes might be helpful for them and also it would help them analyze, communicate and discuss it with sighted colleagues.

The focus of this research is to find the effect of keeping the spatial information of class diagrams; through designing a methodology to navigate accessible UML class diagrams for software engineering/programming students or engineers. We are exploring the new technique of how to navigate the diagram. Diagrams can be represented in different ways, such as a hierarchical structure (King et al., 2004) or a connected network (in other words giving the exact spatial representation of the diagram or an interpretation of it). If choosing the connected network diagram the question would be how to search or visit the nodes. Here we suggest using keyboard keys combination (arrow keys, but we keep in mind that keyboard functionality changes when using screen readers) and numeric keys.

To test this technique a prototype has been built to test the suggested new methodology that is also presented in this research. To complement it, specific rules were also designed for the alternative text that was used with the diagrams. The connected network representation is given with a new technique, which helps users to choose different paths when navigating the diagram. This prototype is tested in a user study to find what strong points are in the design; and how the design could be enhanced, taking suggestions from real possible users in the future. The study tries to explore the satisfaction of the participant with suggested methodology, and the understandability of the alternative text given, and whether the user can build a cognitive map of the diagram.

Other points taken in consideration here are to solve the problem in a way that it would be available for the programmer with visual impairment easily; finding ways that will not need extra tools or hardware that is usually expensive. Also trying to find and explore ways to make the solution available for different platforms and compatible with different screen readers is an essential point. This research is targeting a very specific user demography which includes persons with visual impairment who want to pursue Software Engineering as a profession, or persons who would have some classes in programming.

1.1.1 Why Class Diagrams?

There are different types of UML diagrams; each presents different aspects of a system and has different properties. Our final goal would be covering all diagrams. For this research we used class diagrams. Class diagram was chosen as it is considered the backbone for object-oriented design and the most used type of UML diagrams, and it's the most used diagram in programming and software engineering classes (Müller, 2012). King in his research (King et al., 2004) supported four diagrams out of 12 (then) and 14 now (*OMG UML 2.5*), and they chose the diagrams that they considered the most useful ones; class diagrams was one of them.

1.1.2 Cognitive Maps

This term was first presented by Tolman in 1940s (Tolman, 1948), which can be defined as mental construct which is used to understand and know the environment and which is used then to make spatial decisions (Kaplan, 1973). These maps might be different from the actual environment. Using it a person can plan their movements toward an unseen goal using the available knowledge (Papadopoulos, Barouti, and Koustriava, 2016) (Péruch et al., 2006), so they need to build/form the cognitive map to be able to perform their tasks that involve spatial information. The process of cognitive mapping helps individuals to acquire, store, recall, and decode information about the relative locations and attributes of the phenomena in their everyday spatial environment (Downs and Stea, 1973).

However different terms have been used to convey similar or different ideas (Papadopoulos, Barouti, and Koustriava, 2016) (Kitchin, 1994). There are other terms that have been used in similar researches where the the concern of the researchers is for people with blindness to build these "cognitive maps". Some of these terms that are used are mental representation (Bardot, Serrano, and Jouffrais, 2016) (Ferres et al., 2013), Mental image and mental model (Kamel and Landay, 2002) (Balik et al., 2014) (Goncu et al., 2015), mental map (Kennel, 1996). The context and the work that have been done in all these applies that the use of all these terms was referring to the same point. The term cognitive map will be used throughout the paper.

1.2 Motivation

As discussed in introduction, there is a need for making UML diagrams accessible; and there is a need to understand the best way the UML information will be analyzed, represented and communicated in a way that factual and important information will not be lost. The main objective of this research is to find the effect of keeping the spatial information of class diagrams, and its effect on programmers with visual impairments and their ability to build a cognitive map of the diagram using the provided information. And whether this cognitive map is helping them to read the diagram in a better way. We are exploring the new technique of how to navigate accessible UML class diagrams for software engineering/programming students or engineers. Also we try to find what strong points are in the design; and how the design could be enhanced, taking suggestions from real possible users in the future. The study tries to explore the satisfaction of the participant with suggested methodology, and the understandability of the alternative text given, and whether the user can build a cognitive map of the diagram.

1.3 Research Questions

The study is designed to answer the following questions:

- RQ1: Is the user able to create a cognitive map of the diagram?
- RQ2: Is the user able to identify the different classes and relationships in the diagram?
- RQ3: Is the information provided for each element (class or relation) enough?
- RQ4: Is it easy / efficient to navigate or navigate back to a specific point?

Gathering answers for these questions will help us make solid decisions and finalize the design, and test the effectiveness of the navigation methodology. An initial design for the navigation methodology and alternative text rules are made, and to further explore the research questions, a prototype is built and tested with programmers with visual impairment.

1.4 Contributions of the Thesis

- We propose a novel navigation scheme that preserves the spatial information of the diagram, and allows the user to choose different paths in navigation.
- A prototype to test the navigational scheme
- A user study with programmers with visual impairments, to identify points of weaknesses and strengths and to suggest modifications.

1.5 Thesis Organization

This work is organized as follows: Chapter 2 explores related work in literature. Chapter 3 describes the methodology, where the design of the navigation scheme and alternative text rules is given; then Implementation of the prototype and its choices and challenges are given. In chapter 4 the suggested methodology and prototype are evaluated with a user study, the results are also given with discussion then threats of validity are presented. Chapter 5 discusses future work and a conclusion.

Chapter 2

Literature Review

Related work includes research that has been conducted to make different types of diagrams and graphs accessible, including UML diagrams. While the literature doesn't have a reach content, lots of tools that help with UML diagrams or even solutions that could leveraged to make UML diagrams accessible. We decided to expand the literature review to more topics that can be connected to our research, where we can use and benefit from some experience or results from other researchers, even if not directly related to UML diagrams. For example, even though we are concerned with reading UML class diagrams, we discuss some insightful and helpful experiences in creating UML diagrams or other kinds of diagrams and graph; mainly we look into tools that used spatial representation. We also include in our literature review experience done with other images like floor plans or eBooks. Generally, research related to this one may include surveys, making diagram creation accessible and presenting/reading diagrams.

2.1 Surveys

Some papers are centered around understanding the needs of programmers and students with visual impairments, and to identify areas with difficulties for them. This was of a great help for us to confirm what is used and what is available not only in literature but also in practice for programmers or students with visual impairments.

Developers and programming students with visual impairment still face different hardships and lack of tools to help them advance in this area; personal efforts are done by many of them to fill the gap. Albusays tried to identify the challenges that face developers with visual impairment by conducting a survey with 69 blind developers (Albusays and Ludi, 2016). Some of the difficulties were related to: debugging, UML diagrams, code navigation, inaccessible IDEs, and working in teams. These subjects are broad and different, a further investigation is needed to have a better understanding of each problem. The two areas or problems relevant to this work are problems with accessing or dealing with UML diagrams, and working in teams with sighted people.

Also understanding the basic and essential material that is used by programmers is investigated, to give it priority in making it accessible. If UML diagrams are made accessible what would be the diagram type to start with? A diagram that is widely used and known and needed by most programmers. (Müller, 2012) presents a survey with two blind students in two different universities to show what kind of material they would be exposed in their programming classes. The results showed the computer science lecture had 468 graphics and 120 UML diagrams, while a software engineering class had 425 graphs and 313 UML diagrams, both showed that the most used diagram was class diagram.

They also present a simple way of giving a textual presentation of the UML using tables, even though no deep description and evaluation are given, but it shows that the textual presentation was given with tactile prints, which shows both are needed for the student.

2.2 Reading and Presenting Diagrams

Goncu (Goncu and Marriott, 2015) presents an app called Graviewer an app that can be used on iPad to present accessible graphs and a web-based tool called GraAuthor. Using voice over for reading iBooks will give an alternative textual description that is associated with graphics which is limited. But the new idea is giving the ability to a sighted person (even if not a trained transcriber) to make images/graphics accessible using audio feedback with no need to an expert and add them to the iBook using GraAuther. This content can be accessed from the iBook by double tapping it, where the user will be taken to GraViewer.

The same researchers (Goncu et al., 2015) presents Grafloor an online service that takes floor plans and make them accessible automatically using sonified output and textual presentation and can be viewed using GraViewer. Also manual transcription is available by integrating GraFloor with GraAuthor where GraFloor output can be enhanced in GraAuthor. This research deals with three different problems: graphic recognition, automatic transcription, presentation technologies.

This approach helps users build a mental model (cognitive map) using both hands without relying on expensive tactile graphics, it also presents spatial and textual information. This paper is relevant as they are trying to preserve the spatial presentation of the graph. However it has one main disadvantage, which is the dependency on sighted people to create the accessible content. We take the successful application of preserving the spatial information, while being aware of the difference in the nature of UML diagrams and floor plans. Floor plans have a real physical spatial information, UML class diagrams spatial information could change depending on the creator even if same connections are maintained.

King (King et al., 2004) worked on taking the output of a UML design tool and saving it as XMI (XML Metadata Interchange) and presenting it in an accessible way. Some of the hardships that face researchers are: 1) different design tools and their inconsistency. 2) The big amount of data in the diagrams. Also it's discussed whether spatial information should be given or not. They argue even though it might be needed by a blind developer to communicate with other sighted colleagues, but the effort to give this information is more complicated than needed. Their suggestion is to work around this, and rather than giving the absolute information related to spatial layout, they present what is being inferred from it.

Their suggested diagram representations include connected network where the user moves from node to node using the joystick, the problem with this representation is if there is more than one node in one direction, this representation sparks the idea of presenting the diagram as connected network that can be navigated using keyboard combinations. Another approach is to present the diagram as an internally-hyperlinked text only document. Lastly is an idea to present the diagram in tactile tablet. They chose to represent the diagram in a hierarchical structure in a tool called TeDub.

Kennel (Kennel, 1996) presents a diagram reader called AudioGraf that helps presenting simple diagrams in tactile way. A description of the used attributes is given for frames, text and connections. AudioGraf can be used by both sighted and blind people by having visual and auditory presentation. The blind person can select what to hear using a touching panel. This method helps building a mental map (cognitive map) for the diagram. Two levels are presented, counter where the user specifically selects what to hear on the panel; Focus mode gives a square with working point in center, AudioGraf will auditory display whatever is within the range. Also two views are provided attribute and element views.

Two main points were taken in consideration, firstly the focus on not having dependency on sighted people help, and the second is developing an approach that depends on available hardware which is in this case a PC with soundcard and touch panel. Also the idea of providing multiple options for the user would increase its usability.

2.3 Creating Diagrams

Ishihara (Ishihara et al., 2006) proposed a method to create metadata to describe different relationships between presentation objects. These relations include parent-child relationship, a sibling's relationship based on objects proximity. Arrows also are detected with their source and target and related labels. An "adaptive interface" called DocExplorer is introduced which is an alternative interface that is optimized with screen readers. This interface presents the metadata in a tree view. Also information can be edited using text view and the changes are applied to the original presentation. Other changes can also be applied such as adding new slides or changing slides design. This method is specifically used for presentations and slides, and exploring different aspects rather than focusing on one, where they are introducing methods for presenting metadata, an interface to read it and modify the presentation.

Kamel and Landay (Kamel and Landay, 2002) present a method and a drawing tool that applies the method called IC2D. The method is based on dividing the screen into smaller, navigable work space that consists of 3x3 grids. These nine cells are assigned numbers and are navigable using keypads. As an output the tool gives audio feedback. A user study is conducted with 16 participants (sighted, partially blind and totally blind), the study showed the not sighted people were satisfied with the navigation method and grid interface was intuitive. All participants were able to complete the tasks. Also two long time users were able to draw more complicated drawings. The grid and keyboard navigation method would be helpful with making UML diagrams accessible; as a programmer is expected to be working with a computer, switching to other devices to use UML diagrams would be time consuming. Having a fixed grid as this method might be restricting especially for large UML diagrams, Which is a notable difference with our approach.

Balik et. al. (Balik et al., 2013) built a tool (application) for windows platform called GSK to make graph sketching in STEM accessible to blind students and professionals. The tool is able to create, edit and share graphs. The mechanism used is based on grids and navigation using keyboard. The functionality in the tool is designed in a way that makes using it easy for both blind and sighted people. This is done by providing functionality and navigation using a point-and-click interface which is usually used by sighted people, and keyboard navigation using arrows mainly and having programmatic focus for blind people. Moving between the nodes and the edges will move the visual focus at the same time the name of that component is announced by the screen reader. Another feature that helps this communication between blind and sighted users is keeping the spatial layout of the graph using grid layout. Obviously a main difference from our research is the fact that this is used for creating graphs and not reading them.

The tool was tested with one user who is also a coauthor of the paper to create and communicate different diagrams such as Graph theory, NP-Completeness Proofs and Automata. A study was held (Balik et al., 2014) with ten participants including the co-author, with different expertise and an age range 14-30. The study had different tasks using GSK, and excel was used as a control. The study had three parts, Graph examination study, Graph navigation study and Graph creation study; each part was followed by few questions. Generally all the participants were able to finish their tasks with varied timings and with overall accuracy rate of 99.3%. After conducting the study with the first eight participants, some changes were made and tested with the last two.

PlantUML (*PlantUML*) is a tool that creates diagrams from a simple textual language. Creating diagrams in this case is more like programming, which would make creating diagrams more like "writing" diagrams. PlantUML is open source, which means its available to everyone. It supports 10 different diagrams, and can be integrated with IDEs. The way PlantUML works and its availability makes it a great option or creating UML diagrams for programmers with visual impairment.

Chapter 3

Methodology

Presenting a diagram for programmers with visual impairment could be done using tactile prints, however this would only mean that these diagrams would not be available momently to them and it will cost a lot to print each diagram just to be checked. On a regular bases, a programmer might want to check multiple examples of different diagrams that might be available online for the sake of a homework or a quick design decision; An option that is not accessible by those programmers or students (Albusays and Ludi, 2016). In this section, we present our suggested navigation methodology, alternative text rules and how these suggestions were implemented in a prototype.

3.1 Design

The design includes two main sections. The first section is concerned with designing a scheme for navigating a class diagram, while keeping the spatial information. The second part discusses the given alternative text that will be associated with each element of the diagram.

3.1.1 Navigation Scheme Design

Preserving the spatial information of a diagram is helpful and intuitive (Goncu et al., 2015) (Goncu and Marriott, 2015) (Kamel and Landay, 2002), it might be helpful for persons with visual impairment and would help building the cognitive map of the diagram (King et al., 2004). Diagrams can be represented in different ways, such as internally-hyperlinked text only document, hierarchal structure and connected network (King et al., 2004). Kamel presents a drawing tool (Kamel and Landay, 2002) that is based on dividing the screen to nine cells (3*3 grid) and is navigable by keypad.

The fixed grid given by Kamel might be restricting especially for large UML diagrams. The grid and keyboard navigation method would be helpful with making UML diagrams accessible; as a programmer is expected to be working with a computer, switching to other devices to use UMLs would be time consuming. However a different approach could be given in Class diagram case. Having more grid cells would mean having less information in each one, which is a better accessible design option. The class diagram would be divided into components for each cell, where each component would be either a class or a relation. In order to make unrelated classes not navigable, an empty unnavigable cell would be placed wherever needed as shown in Figure 3.1. Relations are placed in separate cells as they already contain lots of information such as multiplicity values or different notations.

Touch screens could be used (Goncu et al., 2015) (Goncu and Marriott, 2015) in approaches that make other kind of diagrams such as maps accessible, while preserving the spatial information of these diagrams. We preferred here using keyboard; as this would be the natural environment of a programmer and using the keyboard would be a norm for them. The diagram components would be fully accessible by keyboard. The user can enter the diagram using tab key. First he would be landed on the left top corner of the diagram, if empty then the closest one to it. Starting from this point the user would be able to use keyboard arrow keys for navigating different cells of the diagram. While navigating through the diagram Focus mode is on (the selected cell will be highlighted), to help people who might have different levels of visual impairment locate the component if needed, or communicate the material with sighted colleagues if needed.

In the case of reaching a dead point a proper message will pop up indicating that. The pop up message is a temporary solution as sonification is not being studied in this research; however in a complete product sonification would be preferred. Pop up messages will also appear when there are two navigation paths in one direction. The user will be asked to choose a path to continue navigating. This will be done by choosing one of the available options hitting numeric keys in the keyboard.

As the screen reader will be on for a person with visual impairment, alternative text for each element will be read and the user can also use different shortcuts to read the content word by word when needed.

3.1.2 Alternative Text Rules

The notation of UML class diagram is efficient and meaningful for any person who had a basic object oriented programming class. For this reason the alternative text presented for each element uses class diagram notation directly without any further explanations or derivations. The alternative text is designed based on information from (*IBM UML basics: The class diagram*) and (*UML-Diagrams | UML Multiplicity and Collections - defining and using multiplicity and collections in UML - lower and upper bounds, cardinality, order, unique.*) as following.

Each node (Cell) will contain detailed description about the class or the relation that would be read by the screen reader. If the user wants to slowly listen to information in a cell, he would be able to do it word by word or whatever way suitable for them. Specific rules are made to have template that would be followed for elements, and as following:

- Each element starts with the word CLASS or RELATIONSHIP, followed by class name for class and relationship type for relations. This will help the user to know the type of element and what it is quickly, and quickly pass it if it is not what the user wants when searching for specific things.

- For each relation the following sections exist:
  RELATIONSHIP TYPE: "type".
  Two related classes with details.

- Template
  RELATIONSHIP TYPE: "type".
  FIRST END: CLASS "class", ROLE NAME: "role", MULTIPLICITY VALUE: "MV".
  SECOND END: CLASS "class", ROLE NAME: "role", MULTIPLICITY VALUE: "MV".

If inheritance relationship it will have:

Superclass: "class".

Child class: "class".

- Example: a relation from Figure 3.1:

  Relationship Type: Inheritance.

  Superclass: User.

  Child class: Administrator.

- For each class the following sections exist:

  CLASS "Name".

  Class attribute list: with the format

  attribute name : attribute type

  Class operations list: with the format

  name (parameter list) : type of value returned.

- Template

  CLASS, "name".

  ATTRIBUTE LIST:

  "attribute name": "attribute type".

  OPERATIONS LIST:

  "name (parameter list)": "type of value returned".

- Example from Figure 3.1:

  Class, User.

  attribute list:

  User id : string.

  Password : string.

  Login status : string.

  operations list:

  Verifying Login() : bool.

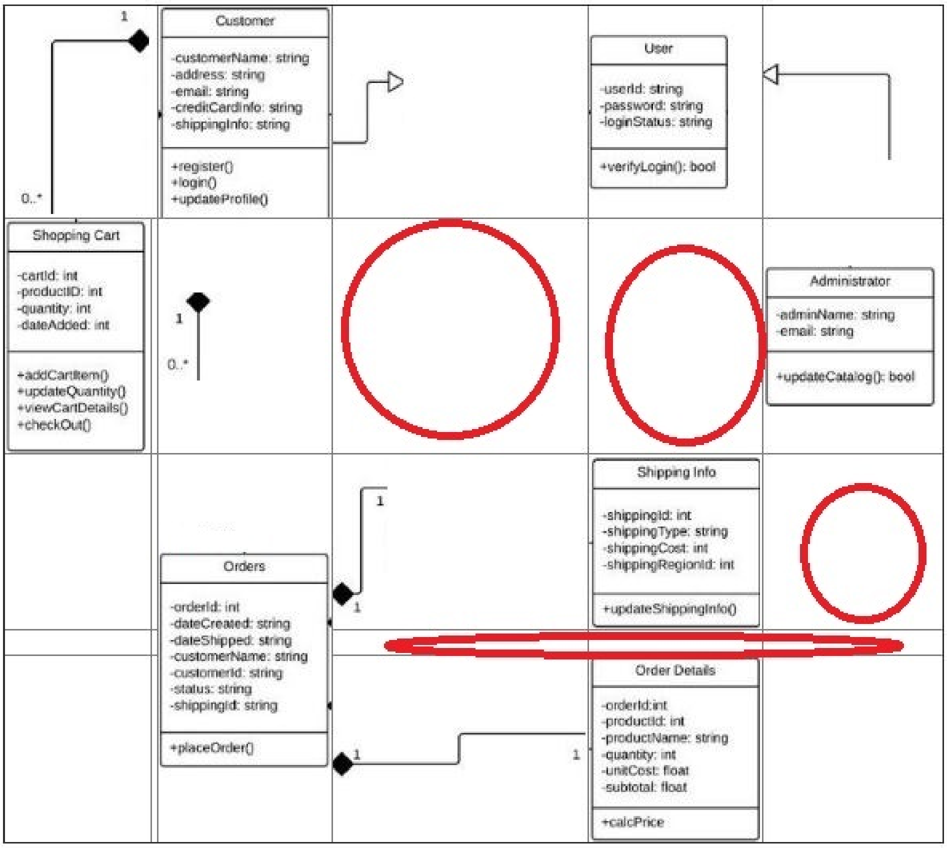

FIGURE 3.1: UML class diagram (*Lucidchart | Class Diagram for Online Shopping System (UML)*) placed as is in the navigation grid, unnavigable cells are marked with red circles.

3.2 Implementation

3.2.1 Implementation Choices and Challenges

The prototype is built to test the suggested methodology using HTML, CSS and JavaScript. No automatic graphic or text recognition is made. The reasons for choosing HTML is for future purposes, as the intention of researchers is to make the product web based; which means testing the methodology with HTML made more sense. Also for easiness of implementation as this work is an evaluation prototype. However problems were faced due to the nature of HTML.

DOM (Data Object Model) of HTML is a standard that defines how to deal with HTML elements, including access, get, change, add, and delete operations on any element. Usually, when loading web pages a DOM of the page will be created as a tree of objects. Dynamic HTML can be created using JavaScript.

For HTML tables that are used as a grid for the diagram in this prototype, DOM affects how navigation using keyboard would be in the table cells. As can be seen in Figure 3.2, after entering a table body, navigation is possible from one row to the other. In other words you can navigate/move cells horizontally, and to go to any cell in the next row, you have to start at the beginning of the next row; which means moving through the cells is sequential and the end of a row would lead to the beginning of the next one.

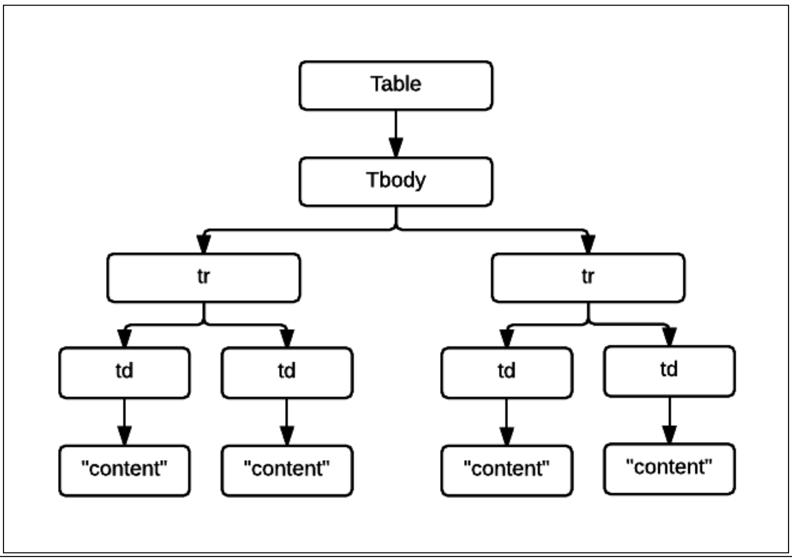

FIGURE 3.2: HTML table DOM object tree.

Other problems also appeared, related to dis-figuration of the table rows and columns. This was due to merging some cells that was needed to imitate the original class diagram to save the spatial information of it; this can be seen in Figure 3.3. A special script was written to show a row and column that a cell belongs to, when hovering over it with mouse.

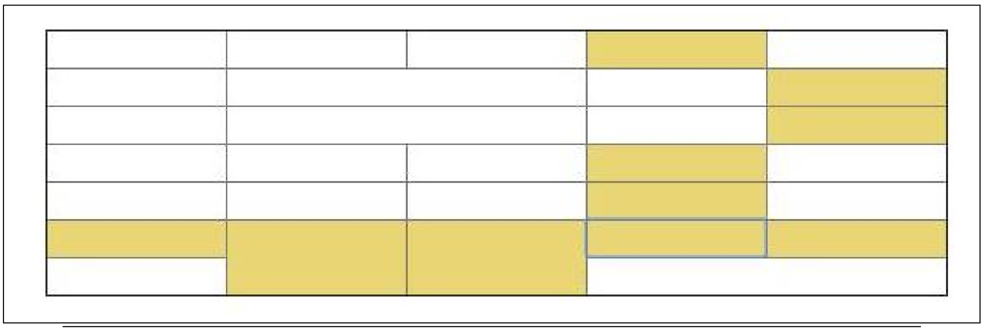

FIGURE 3.3: HTML disordered table due to changes, shown by a script that indicates row and column that a cell belongs to.

3.2.2 Navigation Algorithm

An algorithm and script for decision making in a cell was developed to overcome the default HTML DOM and the disfiguration of table, also to insert and control table endings. The user in this case would be able to navigate the diagram elements as proposed, using Keyboard arrow keys and using numerical keys when choosing between different paths. The script follows the flowchart presented in Figure 3.4.

When the user enters the table (diagram), Focus mode is invoked automatically; using the tab key the user can enter or exit the diagram. When entering the diagram the script will completely takeover and the user can navigate as planned, visiting different connected elements and listening to alternative text connected to each element.

The same reasoning is used when moving in any direction. After clicking an arrow key, the script will check if this is the end of the grid, and will give a proper message when needed. If the current cell in the grid is not in the margin of the table on that specific direction, then the number of triggered cells will be checked. If there is only one cell in that direction then it will check the focus-ability of the cell; as we mentioned before there might be unfocusable cells in the middle of the diagram, to stop navigation between unrelated elements. In this case a proper message will also appear. If the next cell is focusable then focus will move to that cell and the user can repeat the process.

When having more than one element in the targeted direction; if only one is focusable, then focus will automatically move to that cell. If there are more focusable elements in one direction, and that is triggered, the user will be asked to choose one path by hitting keyboard numeric keys. According to that selection, focus will move to the next element, and the process can be repeated.

3.2.3 Technologies Used

While developing the prototype, the main browser used to test the prototype was Google Chrome and the used screen reader was NVDA for windows. NVDA was used as it is free thus more available to be tested. While Internet Explorer IE and Firefox are more popular browsers among screen reader users (*Which web browsers* *are most accessible? | DO-IT*), we decided to go with chrome for many reasons. First, Chrome is most popular web browser (*Browser Statistics*); this trend has been continuous for years which as we see that this would mean more support would be given to this web browser. Chrome also has increasing support for accessibility and also have a good support with developer tools and extensions. Secondly, during implementation and test Chrome was more responsive to accessible guidelines and code. When using these specific technologies arrow keys alone were used for navigation and numeric keys for choosing paths, also focus mode would be invoked automatically in the diagram. The prototype was also tested and altered to be testable with other technologies, to make sure all participants can test in a familiar environment. Mac and VoiceOver worked as intended exactly. When using JAWS, the user would have to press insert+z to enter focus mode each time the page is reloaded.

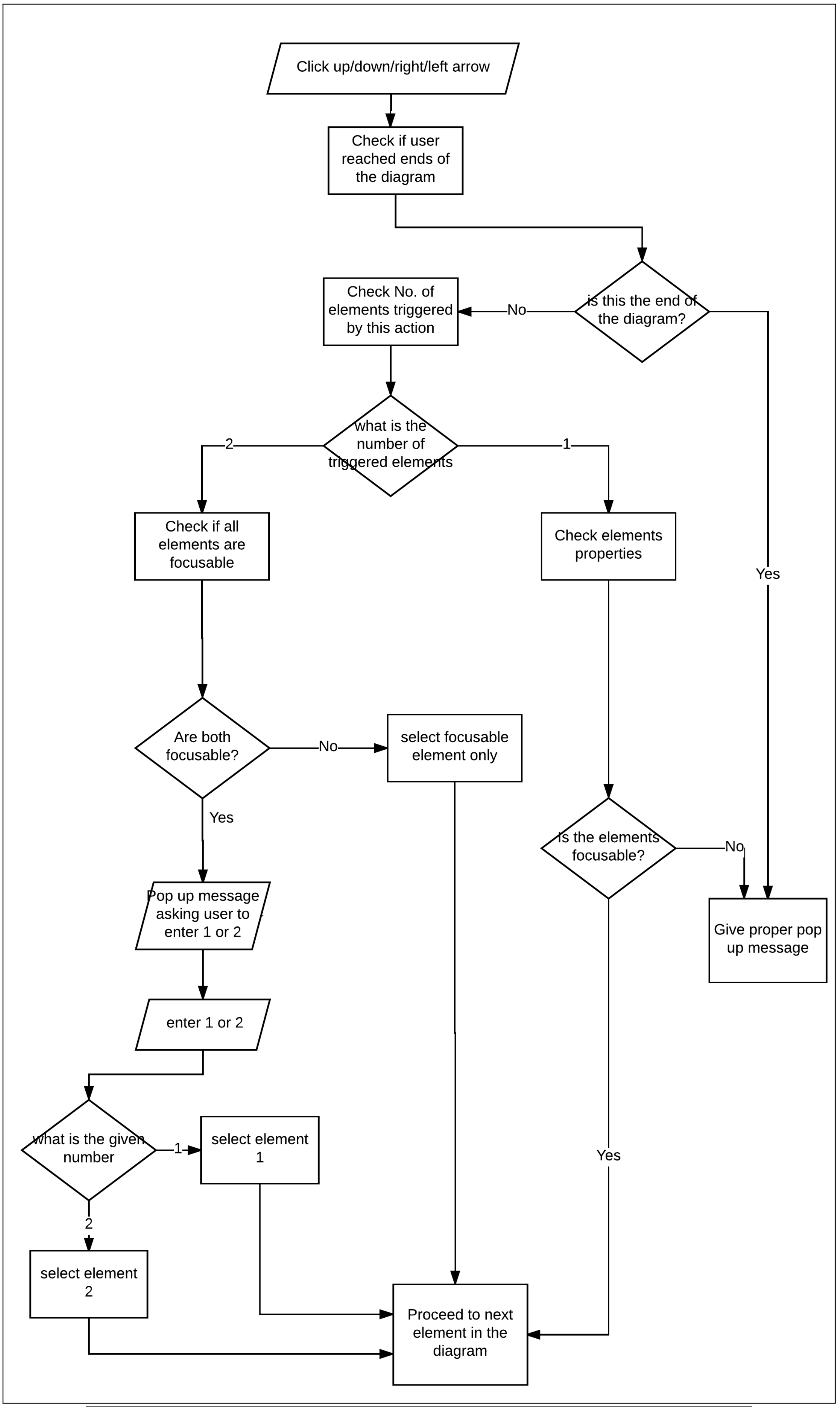

FIGURE 3.4: Flow chart describing the decision making process for cell navigation.

Chapter 4

Evaluation

To evaluate the suggested navigation scheme, the built prototype had to be tested with participants who represent real intended group of users. A user study was conducted for this reason. In this chapter we present the user study design and details. Then we present the results that we got, and discuss them in the light of our research questions, and what enhancements can be suggested to the given navigation scheme. In the end we discuss threats of validity to this study.

4.1 User Study

The user study mainly tested the navigational functionality of a class diagram using keyboard with a focus on multiple path choices for programmers and students with visual impairment. This was intended to help to understand whether the proposed diagram representation is helpful to build a cognitive map for it, and discover the different elements in the diagram and the connections between them. This user study can be considered an exploratory study.

The user study was mainly designed to be a remote evaluation of the product. As the study targets very specific demography, there would be a hardship in finding enough participants locally (Petrie et al., 2006), or participants who fit the criteria needed for the specific study. While such potential participants could be found in different places; traveling to them would be an option that will be expensive. Recruiting blindfolded participants (Yoshida et al., 2011) was not an option, also we noticed during the pilot tests that the blindfolded person is handling the situation from a sighted person point of view and it was giving unrepresentative results. For the user study that was done with the prototype, a remote study was conducted.

However remote studies have shown their success in many previous studies if correct tools and measurements are used.

4.1.1 Participants

For testing the prototype three participants that fit the inclusion criteria were recruited. The participants were found through a special mailing list through sending a recruiting email. Participants were screened based on their knowledge and familiarity with object oriented design and UML class diagrams. Two more participants were fit to participate in the study, after filling the background questionnaire. One of them couldn't participate later, and the other participant's session was canceled due to different connection reason; both participants background questionnaire information were taken in consideration for the results.

Some of the general conditions were the participant age, no participant less than 18 years old was accepted; according to IRB conditions, persons less than 18 would need more caution and different arrangements. The three participants' age range was 25 to 31. While the participants with different visual impairment degrees would have been included, the three participants were all totally blind as they identified. Meanwhile one of the two canceled study participants was totally blind and the other had a low vision with screen reader usage. All participants were males.

As inclusion criteria, the participant must have had at least one object oriented course. This was a crucial point, so the participant would have understanding of what he/she is exploring. All had degrees in Computer science. One had 1 to 5 years' experience and the other two had 6 to 10 years. The two participants who couldn't take place in the study both had experience more than 10 years with programming. Results give an experienced programmer point of view according to the previous information. Having students as participants might give slightly different results. It was expected by the researcher that experienced programmers might have not used UML class diagrams for a while, hence forgetting details about it; however all participants answered positively to using class diagrams and showed no lack of remembering class diagrams.

4.1.2 Study Environment and Setup

The evaluation was conducted in users' normal environment and their own machine, to have a natural set since they have familiarity with their own system; this helped us see the user in their context (Rubin and Chisnell, 2008). For later analysis, the session included recording participants' screen and voice information while conducting the study. This would help analyzing the recorded material and having a better understanding of how they navigate the diagrams. The recording was done in the evaluator machine, while the user sharing their screen with them using Skype call. Skype calls have an option of sharing screen. Recording the study was done in this way for convenience, so the user will not have to install multiple programs on their machine. The study was a synchronize study which means the evaluator led the study and was available to help the user with minimal intervention (Petrie et al., 2006). The Study took less than one hour for each participant.

4.1.3 Study Instruments

Prior to the session by few days a questionnaire (Appendix C) was shared with the participants to collect different information including:

- Their age (as mentioned in the previous section for inclusion criteria).
- Basic programming experience and familiarity with object oriented design and class diagrams. (This was also used as inclusion criteria).
- Tools they use when dealing with UML or UML class diagrams. (This information was used to have a better knowledge about the way the participants deal with UML class diagrams).
- Their computer environment (operating system, browsers and assistive technology). (This information was needed to check what technology the participants use).
- Informed consent.

The informed consent was combined with the questionnaire for ease of use and to limit number of emails sent back and forth. Any user who would not fit in the study criteria was not going to get to the step of signing the informed consent; Qualtrics (*Qualtrics*) a survey software gives the option of having different paths according to chosen answers.

After the end of the study the participant was given a post study questionnaire (Appendix D). The post study questionnaire contained Likert scale questions and open ended questions.

Qualitative data was collected in a semi-structured debriefing session. Also comments from the participants were taken during the sessions or after performing tasks, this was done to allow the participant to convey his comment in the best way while he still can remember the details of his experience. The debriefing session comments and the answers that were taken from the open ended questions in the Post study Questionnaire after the user has finished and got an overall experience.

Quantitative data was collected during the sessions. The data that was collected included:

- Number of tasks completed correctly including the tasks with given assistance.
- Number of tasks completed correctly without assistance.
- Time required to finish a task

These performance measurements are collected according to specific rules. We tracked number of tasks completed successfully with assistance and without it, with no specific time benchmark expected. To be consistent, assistance was given to participant only when they report their final answer to the task with missing or wrong information. Also verbal or questions related to the task wording itself were not considered assistance. The type of assistance given was to tell them to look more as the answer is not complete (Rubin and Chisnell, 2008). For task completion time also we have had an operational definition which gives the procedural details of how this measure is calculated. The start time was calculated with the end of reading the task, the timing ends when the participant started answering with the right information. The timing would be stopped only if technical issues occurred.

4.1.4 Study Procedure and Tasks

The Study consisted of 4 sessions, two training sessions, one main session which included navigating a class diagram as designed to maintain the spatial information, and a fourth session which included navigating a textual representation of another UML class diagram. The last part was added to the study to have a comparative result for some aspects that are being tested. Below is the general session outline:

- Pre-test arrangements:
  - Fill out background questionnaire.
  - Review and sign non-disclosure and recording permission.
- Configurations and recording system:
  - Send study material to participant.
  - Call using Skype.
  - Sign non-disclosure and recording permission, if not done earlier for any reason.
  - Connect to participants screen using Skype's screen share.
  - Start recording using Camtasia Studio Recorder.
- Introduction to the sessions:
  - Start reading the orientation script.
  - Discuss participants experience regarding UML diagram (depending on their answers on the background questionnaire).
- Training sessions:
  - Navigate through the first diagram (Training).
  - Navigate through the second diagram (Training).
  - Discuss with the user if they understand how to use the proposed system.
- Main Session
  - Part A: Navigate through the diagram using tasks.

  – Part B: Navigate the textual representation diagram using revised tasks.

- Debrief session. Ask Questions to collect preference data and qualitative data
- Post study questionnaire

Training Sessions

A fixed narration/orientation script was used with the training sessions to ensure that same instructions and information were given to everyone who participated in the study. First session was a straight forward small diagram, where a user can move horizontally only as shown in Figure 4.1. In this training session, the participant is introduced to basic concepts in the navigation schema, such as what keys to use, what information to expect and what would happen if he reaches a dead end. The second session had more complicated diagram, hence information as can be seen in Figure 4.2. Here the concept of different paths was introduced, and how to deal with it by choosing which path to go through and what keyboard keys are needed. The moderator didn't end these sessions until making sure the participants are familiar with the introduced navigation scheme.

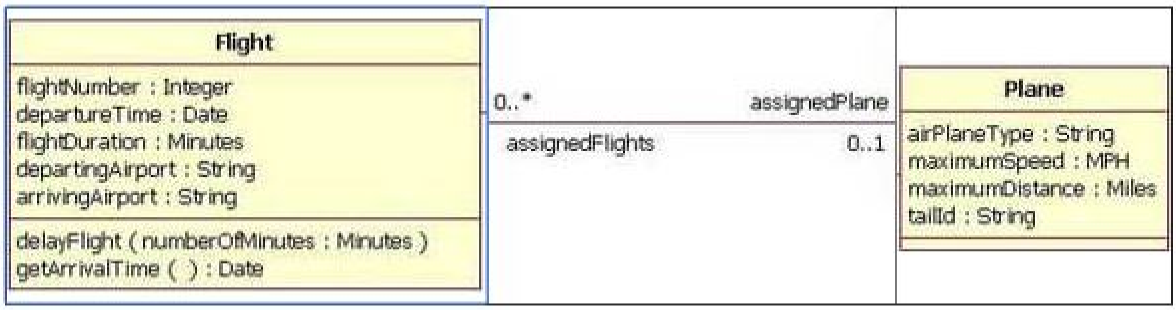

FIGURE 4.1: First Training Session Diagram (*IBM UML basics: The class diagram*), It can be seen that Focus is on Flight.

Main Session: Spatial Info Diagram

The main session included a bigger and more detailed Class Diagram as shown in Figure 3.1. This session included asking the participants to do these five tasks:

- T1: Find class with the name "orders", describe what it contains.

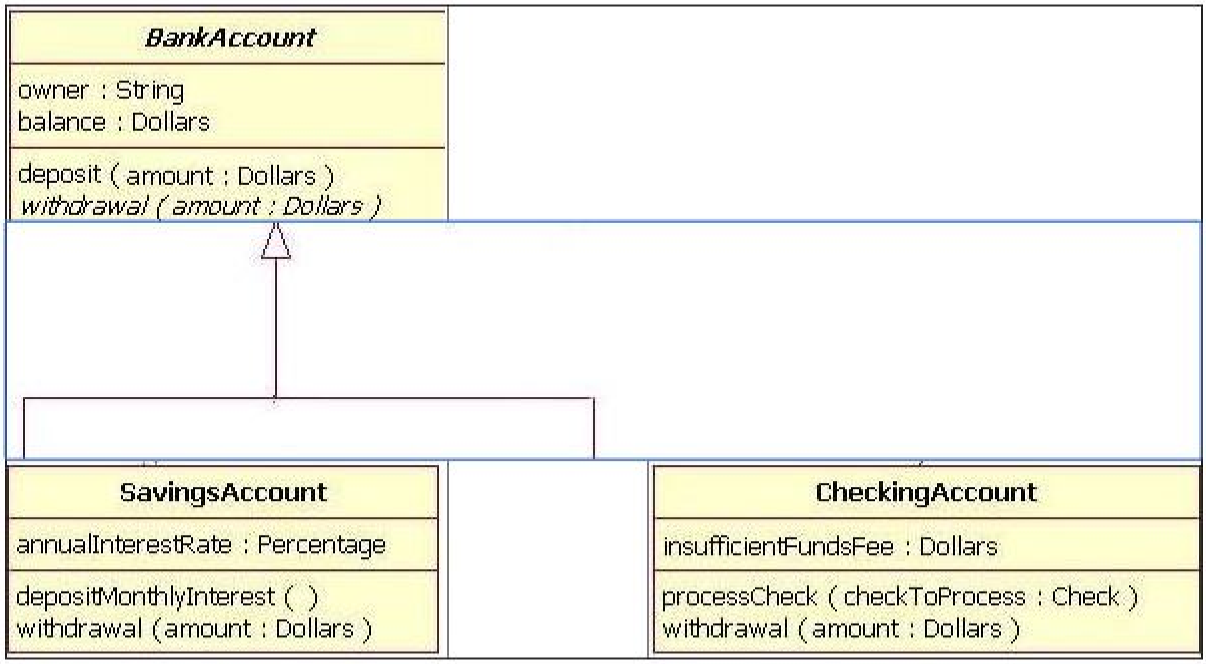

FIGURE 4.2: Second Training Session Diagram (*IBM UML basics: The class diagram*), It can be seen that Focus is on the inheritance relation.

Success criteria: Finding the class with name "orders", and describing its contents.

Research Question mapping: RQ3, RQ4.

- T2: Find the classes that are related (connected) to class “orders”, give their number and names.

  Success criteria: Finding all classes that are in a direct relation with class "orders", and identify them. This result will be compared with the textual representation diagram.

  Research Question mapping: RQ1, RQ2, and RQ3.

- T3: How many classes are there in the diagram?

  Success criteria: Give the accurate number of classes in the diagram.

  Research Question mapping: RQ1.

- T4: Does the diagram contain any inheritance relationship? If it exists, which classes are related with this relation?

Success criteria: Find the inheritance relationship, Identify the classes that are related with this relation. This result will be compared with the textual representation diagram.

Research Question mapping: RQ1, RQ3.

- T5: What is the spatial position of the administrator class?

  Success criteria: Identify the spatial position of the "administrator" class.

  Research Question mapping: RQ1, RQ4.

Main Session: Textual Info Diagram

In order to have a better understanding of the time and accuracy that are measured, a textual representation of another diagram is tested as the fourth and last session. The same elements description (alternative text rules) is used in this representation, arranged as a one dimensional list, which means no spatial representation is given. A comparison between the two results is then analyzed. The following three tasks were given to the participants:

- T1: Find class with the name “Customer”, describe what it contains.
- T2: Find the classes that are related (connected) to class “Customer”, give their number and names.
- T4: does the diagram contain any inheritance relationship? If it exists, which classes are related with this relation?

Task three was eliminated as the textual representation was given as a list, which means knowing the number of classes will only need moving down and counting classes. The fifth task was also eliminated as spatial information is not given here.

To eliminate the effect of different factors that might lead to incomparable results, the two diagrams were chosen carefully, both diagrams have seven classes and both have inheritance relationship. The diagram for this session is shown in Figure 4.3.

The participant was also allowed to give any comments, notes and feedback during the sessions, instead of waiting until the end; this was done to take input from

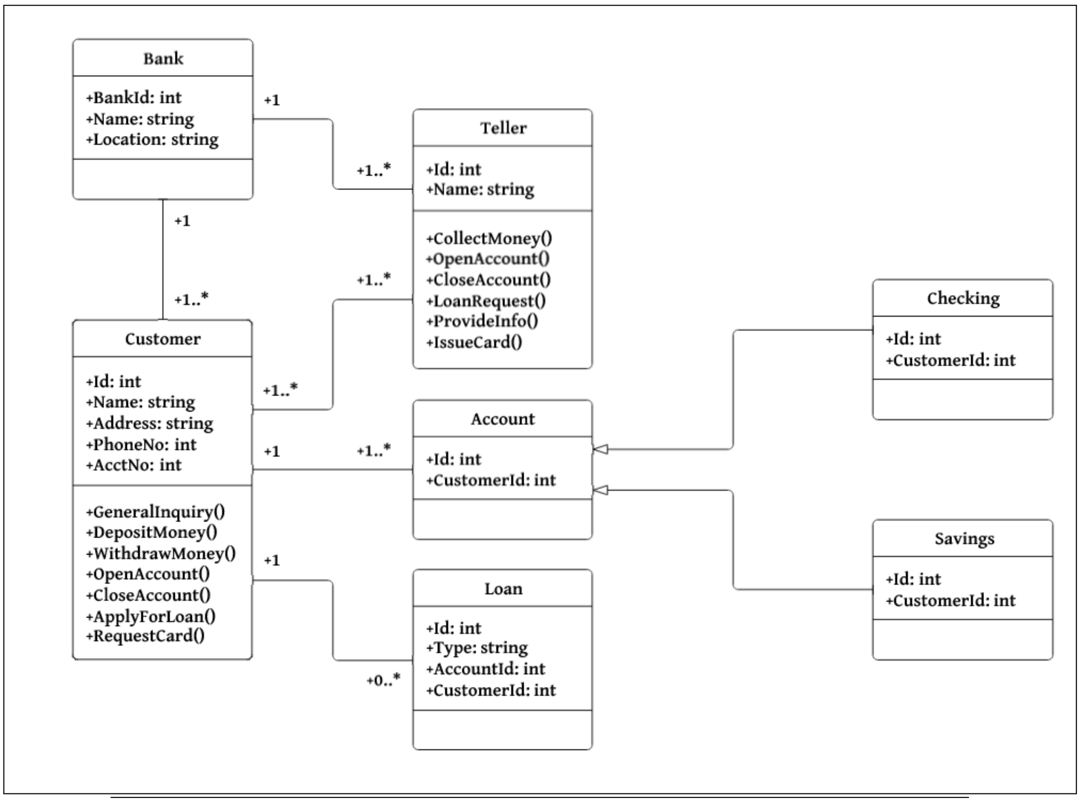

FIGURE 4.3: Main Training Session Diagram, Textual Representation (*Lucidchart | Class Diagram Tutorial*).

them as much as possible when the idea is fresh in their mind. Also the testing session was concluded with a debriefing and a post study questionnaire that contained a likert scale and free-response and open ended questions.

4.2 Results and Discussion

The results of this study can be divided into three categories: background questionnaire, user study session and post-study questionnaire.

4.2.1 Background Questionnaire

As mentioned previously, five participants filled the background questionnaire with the intention of participating in the study. Even though all the five met the participation criteria but only three were able to participate. Getting data from the background questionnaire, two main questions were asked that were not used for screening. These two questions were asked to see the participants' background regarding dealing with UML class diagrams.

The first one was "How do you create UML diagrams". One participant answered PlantUML (*PlantUML*), and another mentioned that he know he can use PlantUML, but never had to do so. The other solution was using text, one mentioned text document, words and wiki sticks, and the other mentioned using text-based program outlines for program design purposes. While searching for different related research in this area, we found PlantUML which seems like a project that could help with our future work.

The other question that was asked was "How do you read existing UML class diagrams?". Most answered that they don't read UML diagrams. One participant mentioned PlantUML script or a coworker; when asked how PlantUML is used for reading diagrams despite being a tool for creating diagrams, he answered "Ask people to Write down the script with description of the diagram, and then have it in a Java file." . Another one added that "the accessible drawing tools for blind did not seem compatible with UML formats". These answers were not surprising as Albusays (Albusays and Ludi, 2016) mentioned such hardships for programmers when it comes to UML diagrams generally.

4.2.2 Study Sessions

In this section we present the results of the study sessions. In these results percentages will not be given as the number of participants is too small, and having percentages will not give a clear image of the data. While a two sample t-test would be helpful to calculate the statistical significance of the two different sample of results (spatial vs. textual), due to the small sample size the results will not be helpful in all cases; we decide to analyze the data and find trends with it. The results and comments we had from the participants were insightful and helped us derive some general trends and modifications to the suggested navigation scheme that can be made to the original design.

Below we will be presenting the results before analysis and relating it to our research questions. For T1, T2 and T4 we will present the results of both diagrams, spatial navigation and textual representation together. No benchmarks were used as the comparison with the textual representation was made.

TABLE 4.1: Study Sessions Results.

Main Session	Tasks	Mean completion time in minutes	Participants completed the Task without assistance	Participants completed the Task with assistance
With spatial information	T1	1:25	3/3	3/3
	T2	3:25	2/3	3/3
	T3	1:52.33	3/3	3/3
	T4	0:12	3/3	3/3
	T5	0:19	3/3	3/3
With textual information	T1	0:14	2/3	3/3
	T2	3:38.33	1/3	1/3
	T4	00:59.67	2/3	2/3

As can be seen in Table 4.1, all participants were able to complete T1 (T refers to Task) for both navigation and textual representation diagrams. In the textual representation one of the participants used search and hence it was considered as an assistance. Less time was needed for the textual representation as it is straightforward and only requires moving down the list to find the required class.

For T2 both approaches took lots of time with an average of 3:25 minutes for spatial representation and 3:38 for textual information. In spatial representation, all participants were able to complete the task successfully with one of them getting assistance. We should also mention that another participant was assisted in this task; the participant knew that class order had two other classes related to it from the right, but he thought both relations are giving same information. As this assistance was not related to spatial recognition but to textual rules, we didn't consider this as a task completion with assistance; but we will mention and discuss this in evaluating the textual description rules.

In the textual representation one participant was not able to answer correctly even after getting assistance, and another got assistance before giving the full correct answer. Both were told that they are still missing some information after informing the observer that they are done with the task. One of them was still missing one connected class in the end, even with the use of search function for the required class name. The reason why the moderator didn't ask the participant to not use the search option was if a regular list could be searched, and a regular user would use it then the scenario should remain realistic. “Textual diagram was handled well because of the find function”, this was the participant’s comment even though he knew he is still missing one related class; which indicates the participant need for the luxury of having this function whenever needed.

For the third task, all participants completed the task in spatial representation diagram without any problems; they spent some time even after knowing the number to make sure they have found all the classes. This is why it took them an average of 1:52 minutes to finish the task.

The fourth task was asking the participants about the inheritance relationship and what are the classes that are related with this relation if it does exist. In the spatial representation diagram, all participants recalled having it in the diagram as soon as they were asked with one of them giving the complete correct answer directly; the other two re-navigated the diagram to make sure of the information and classes names. The average time to complete this task in this representation was 0:12. In the textual representation, only two participants completed the task. The third one found the information but misunderstood it. He mentioned "Account is parent of checking, and checking is parent of savings". This couldn’t be related to the alternative test rules, as same rules are used in both diagrams.

Task five was also completed successfully, with almost no time needed to answer. In the end the participants seemed really familiar with the diagram, remembering where to find things and re-navigating to the answer spot only to recall the exact names and information as they expressed and as noticed by the moderator.

For the alternative description rules that were included for each element, it looked very intuitive and self-exploratory. Two notes were made from observing the participants and from their own comments that can help in making enhancements. The first note was the use of the word “operations”, while this word was used in both resources that were used to extract these rules; both participants had problems with it. Both of them mistook it for relation, and the moderator had to make that clear. The second note was related to relationships. While the rule makes it easy to know the type of relation, the user will have to wait till the end to know what the other end of the relation is. An optimization of the text could be done. An example could be instead of saying “multiplicity value “and giving the value, the value could directly be given. This might not be clear for beginners, but an option of optimized alternative text could be given.

RQ1: Is the user able to create a cognitive map of the diagram?

Here we will be discussing the results from T2, T3, T4, and T5. In this research question we try to find if the participants were able to use the spatial information to create a cognitive map of the diagram, and whether this helped them in their tasks.

In T2 we will discuss and compare the results of both spatial and textual diagrams (find classes related to order "in spatial diagram" and customer "in textual diagram"). We can notice from Table 4.1 two main points. First the average time difference in the two representations is not big with spatial being 3:25 minutes and textual being 3:38 minutes. Secondly, the mere information is not enough, since two participants in textual diagram couldn’t give a correct answer even after being told that they are missing information. Which indicates that diagram with spatial information was better in this case. In diagram with spatial information, participants/user need to search in a very specific area to find related classes, however in a list like design this information might be scattered and the user would have to search the entire list and listen to all information to find related classes. We mention again that one participant used search function and was still not able to find the correct answer in textual representation diagram.

For T3 which is a task in diagram with spatial information only (how many classes are there in the diagram). All participants were able to find all the classes in the diagram without any assistant. The average time needed to complete this task was 1:25 minutes (min 0:13 and max 3:16). As we will discuss in RQ3 results, this task was not included in the textual representation as it would be a matter of going down the list and counting the classes. The Goal of the task was to determine the ability of participants to navigate the diagram and find all classes.

T4 task came after T3, where the participants at this point have traversed the entire diagram. It seemed that participants had built a cognitive map by now. It took them an average of 0.12 only to recall the exact information and all of them remembered the existence of inheritance relationship. One participant gave an accurate answer with 0:0 time. On the other hand the average time for this task in the textual representation was 00:59.67 minutes; which is higher than the spatial representation. One participant found the relation and the related classes but misunderstood it. He mentioned "Account is parent of checking, and checking is parent of savings". This couldn't be related to the alternative test rules, as same rules are used in both diagrams as we mentioned before.

Finally T5, which is a direct question about the spatial information (what is the spatial information of administrator class), all participants were able to complete the task without assistance and with an average time of 0:19 seconds.

Discussing all the tasks, we conclude that not only the spatial information is helping the users build a cognitive map, but also this is helping them in identifying different relations in a better and clearer way unlike a diagram with non-spatial information.

RQ2: Is the user able to identify the different classes and relationships in the diagram?

We found that answer to this research question can be found in RQ1 and RQ3.

RQ3: Is the information provided for each element (class or relation) enough?

Here we will be discussing the results from T1, T2 and T4, however we will be focusing on whether the alternative text was helpful or not to complete the tasks; we will also present and discuss any comments the participants had regarding it.

From T1 "finding class orders and describing its contents", the participants were able to identify it as soon as they reached it, which means that the important needed information was placed first (Walker et al., 2013). Further information was listed after and the participants were able to understand it well. One word was not quite clear which is "operations", where two participants wanted to confirm the meaning of it; it was mistaken for classes or relations. The reason behind it might be the familiarity of people with other terms such as "functions" and "methods". This word was chosen as it is more generic and not related to a specific language (*IBM UML basics: The class diagram*), but can be changed to go with what word is more trendy among programmers.

For T4 (does the diagram contain any inheritance relation...), same as T1 it showed us that participants were able to recognize classes from relations, and the type of relations. One participant stated also the other type of relation that was in the diagram, which is composition without being asked. However One participant during T2 (Find the classes that are related to class orders), couldn't recognize that orders class has two classes connected to it from the right, even though he was aware that there are two paths (Figure 3.1). This mistake happened as the two relations were the same and only the class name from the second end was different. Except for inheritance which the relationship type is followed by related classes, other relations might contain lots of unwanted information. An extra line could be added as following:

RELATIONSHIP TYPE: “type”.

[BETWEEN "class" and "class"] or [FROM "class" to "class"]

FIRST END: CLASS “class”, ROLE NAME: “role”, MULTIPLICITY VALUE: “MV”.

SECOND END: CLASS “”, ROLE NAME: “role”, MULTIPLICITY VALUE: “MV”.

If inheritance relationship it will have:

Superclass: "class".

Child class: "class".

Other than that, one participant commented "the info of each element is quite long, so you can't remember it if you hear it in one piece". However having the relations alone was a decision to handle the amount of information a relation might have. Further optimization might lead to loss of information.

RQ4: Is it easy / efficient to navigate or navigate back to a specific point?

Here we will be discussing results from T1, and T5. Here we will not discuss these tasks in the light of cognitive map and recalling the spatial information, but the efficiency of moving in the diagram. We take in consideration T1 (finding class orders and describing its contents), which includes navigating the map before constructing the cognitive map; and T5 (what is the spatial position of "administrator class), which is navigating back to a place that was visited before at least once.

As we mentioned before in reporting the results, performing T1 needed less time in the textual diagram (average of 0:14 minutes for textual and average of 1:25 minutes for spatial), which was expected. While we didn't have a bench mark here, this time is expected to increase as the diagram increases. For T5, The average time to complete the task decreased by more than one minute. One of the participants didn't actually need to navigate back to answer the task, even without it the time to navigate to that specific point has decreased by a good amount of time.

As this was a prototype to study the effect of spatial information, we found that the user can use it to navigate to specific wanted points in big diagrams (7 classes in this specific case). However, sequential access to classes might not always be efficient when targeting specific points in the diagram or when finding the general structure in the diagram. In a working project, this design can be mixed with a class list (maybe hierarchical), something like a left panel that would give information such as the number of classes in the diagram that can be collapsed to be 2D (spatial) for more details. Some of the comments regarding this point were "If the table is very big, the last thing you want is to traverse the whole diagram" and "If you represent something it is better to have it in Layers, somewhere where you have basic object (some kind of Summary)".

4.2.3 Post-study Questionnaire

Likert scale was used to collect some information as can be seen in Table 4.2, one user strongly agreed with all the statements while the other one gave more neutral-disagreeing answers and the last one gave agree- neutral answers. For the small sample number, giving means or medians and even mode will not be helpful in this case. The results of the Likert scale confirm our findings in our research questions. The overall intuitiveness of the system got a score of 1 to 3, which is reasonable as the participants were able to handle the system and solve tasks. "Finding different components of the class diagram was easy" also got similar results which also makes sense as according to the sessions results, all participants completed the tasks in the spatial representation diagram better than the textual. For "Using the keyboard for navigation to find a specific component was helpful" also as we discussed before, finding specific points would be more reasonable in a 1-D list like format.

TABLE 4.2: Results of Likert scale questions, on a scale of 5, 1 being strongly agree and 5 being strongly disagree.

Field	**Strongly agree**	**Somewhat agree**	**Neither agree nor disagree**	**Somewhat disagree**	**Strongly disagree**
Overall, using the system was intuitive.	1	1	1	0	0
Finding different components of the class diagram was easy.	1	1	1	0	0
Using the keyboard for navigation to find a specific component was helpful.	1	0	1	1	0

TABLE 4.3: Results of open ended questions.

What did you like about the system?
Multi-dimensionality.
It allowed spatial and item by item navigation
I liked how easy it was to identify classes which were related to a class. Also it was beneficial to hear when classes were not related such as in the "Shopping Cart" "Order" class.

All participants gave answers to some open ended questions. Regarding what they liked, one said "item by item navigation" the other said "how easy it was to identify classes which were related to a class" and "you can know when classes are not related". Other comments that were made during the sessions are:
"easy to read and understand and can interact with these elements".
"Haven't been easier to navigate a diagram".
"It makes it easier to navigate quickly".
"I think navigation with the arrows is a good option".
"2D-3D diagrams representation are better".
"In a list, I'll have to check every item in the list to see if it is related to the question or not (or to what i want), for example the relationship to customer".

One common comment that they made was adding a tree view representation, not a separate diagram but as a complementary addition for navigation; “Along with this list view having the ability to jump to the respective position in the diagram from the list view would be very helpful” said one participant. Some participants comments were:
"Along with this list view having the ability to jump to the respective position in the diagram from the list view would be very helpful".
"getting an overall idea about a complex diagram using the keyboard only could be challenging, since it only allows sequential access to its items.".

A comment that also stood out and was coherent with the navigation problems that the participants had, was about knowing information before choosing a specific path. If the user is able to know what is coming ahead, they might choose a different path. There is one problem with this, as mentioned also by participants, each node already has lots of information; adding such extra information to the node would make it huge, imagine a node having its own attribute and operations (this information is already included) and five other classes related to it in different directions. Another option is including this information in the pop-up message. Instead of asking the participant to choose 1 or 2, adding the class name **"choose 1 for 'class name' or 2 for 'class name' "**. Some of the comments regarding this were:
"Possibility of reviewing the names of the sub-items before accessing them".
"Entering into cells to select part of the data only".
"If the user can “open” the relation to see what they are before navigating to that part or going to that path".

We will also mention participants' comments regarding the popup messages (the non-assertive alerts), even though we mentioned in the design section that this choice was made for prototype purpose as this is a whole different area of research, and it is not recommended for any final design.
"If it is Words to be heard without pop-up it wouldn't matter how many times you visit them it will be good".
"Change the dialogues to help balloons. It spares a few keystrokes and some nerves

as well".

Would like to change: "non-intrusive alerts when you reach the limits of the diagram".

4.3 Threats to Validity

Limitations of this work include evaluation limitations. The more participants for evaluation the better. It was hard to find participants for the study, with the criteria of a person with visual impairment and knowledge of programming. We were able to do the study with three participants in the time of study that we had. Also the age range was from 25 to 31, which means the results might be different with other age groups. We can mention that we still had good amount of information and the age of participants was balanced as they are close to college students age, and also had some good experience working in programming.

Another threat of validity is choosing class diagrams. The results and insights of this study might not be generalizable on all kinds of diagrams, depending on the nature of the diagram.

Finally, having the remote study was forced on the study in this situation. To mitigate the side effects of it, we tried to have the connection live; sharing screens was also helpful to have the moderator be aware of everything happening with the participant.

Chapter 5

Conclusion

5.1 Conclusion

In this research we have presented an exploration of a modified way of navigation scheme for UML class diagrams. This methodology gives a new approach of dealing with diagrams element by element, to help create a cognitive map of the diagram by giving the spatial information of the diagram. While previous researches have neglected this specific scheme for various reasons or was not used for reading the diagrams, despite their acknowledgment of the need for the spatial information. We suggested this methodology, taking in consideration many possible obstacles; we also suggested using only computer and keyboard as this is a regular programmers' environment, to reduce cost and make it efficient. The user would be able to use arrow keys for navigating the diagram, and numeric keys for choosing different paths when needed. Also rules for alternative text for different components in class diagrams were suggested to be tested along the navigation scheme.

To evaluate this navigation scheme a prototype was built for this reason and used in a user study. A comparative textual representation was tested a long side the prototype instead of having time benchmarks. The results showed the ability of users to build a cognitive map of the diagram, and perceive different connections and relations, while it was not straight forward to reach a specific point directly. Also it was hard for participants to perceive and find all the classes that were connected to a class, when only textual information was given. A modification could be done by getting strength points from the textual representation and add it to the suggested navigation scheme that includes spatial information. This could be done by having a tree view for quick navigation to enter specific wanted locations in the diagram; which could serve like a quick search for the diagram, from that point the user could search the surroundings of that element.

5.2 Future Work

The research intends to make existing UML class diagrams accessible to programmers and students with visual impairments without the intervention or dependency on sighted users. A suggested system would be to take existing UML diagram images in any form and make them accessible. To implement that, image processing techniques could be used such as Optical character recognition (optical character reader, OCR) for text recognition, and other techniques to detect the different elements (nodes and links) in the diagram. The result of the user study could be used to modify the design before implementation.

More questions also needs to be answered such as, how big can the diagram be, and still the user would be able to understand it. Other features can be added to the design, such as counting/knowing the number of visited nodes. This will help the user to be informed about the coverage area of the visited nodes and links when needed. Also a good feature would be search option; where the user shall be able to search for specific property, such as inheritance, and navigating that specific area if needed without going through the whole diagram to reach an intended node.

Appendix A

IRB Approval

R·I·T

Rochester Institute of Technology

RIT Institutional Review Board for the
Protection of Human Subjects in Research
141 Lomb Memorial Drive
Rochester, New York 14623-5604
Phone: 585-475-7673
Fax: 585-475-7990
Email: hmfsrs@rit.edu

Form C
IRB Decision Form

TO: Silva Matti

FROM: RIT Institutional Review Board

DATE: **February 15, 2017**

RE: **Decision of the RIT Institutional Review Board**

Project Title – Preserving the Spatial Representation of Accessible Class Diagrams for Visually Impaired and Blind VIB Developers

The Institutional Review Board (IRB) has taken the following action on your project named above.

☒ Approved, no greater than minimal risk

Now that your project is approved, you may proceed as you described in the Form A. **Note that this approval is only for a maximum of 12 months; you may conduct research on human subjects only between the date of this letter and February 15, 2018.**

You are required to submit to the IRB any:

- **Proposed** modifications and wait for approval before implementing them,
- Unanticipated risks, and
- Actual injury to human subjects

Return the Form F, at the end of your human research project or 12 months from the above date. If your project will extend more than 12 months, your project must receive continuing review by the IRB.

Continuing review of research and approval of research studies is required so long as the research study is ongoing, that is, until research-related interactions and interventions with human subjects or the obtaining and analysis of identifiable private information described in the IRB-approved research plan have been completed.

Investigators are responsible for submitting sufficient materials and information for the IRB to meet its regulatory obligations, and should follow the institutional policies and procedures for continuing IRB review of research that are required by HHS regulations at (45 CFR 46.103(b)(4), 45 CFR 46.109(e), 45 CFR 46.115(a)(1)) as appropriate to the research activity.

____

Heather Foti, MPH
Associate Director
Office of Human Subjects Research

Revised 02.09.2011

Appendix B

Recruiting Email

Hello,

My name is Silva Matti and I'm doing my masters in Software Engineering in Rochester Institute of Technology.

For my thesis, I'm working on making class diagrams accessible for visually impaired developers/programmers, and your feedback for a new approach that I developed. If you have taken any object-oriented programming courses or are majoring in Computer Science or a related field, you are eligible.

The study will involve using a simple website designed for this purpose, and answering some questions while recording your screen and voice only, and will be done some time between April and May. Session might last about 60 minutes. If you participate, your name will be entered in a raffle to win $50.00 gift card.
If you are interested, please contact me.

My Best Regards,

Silva Matti
MS student | Software Engineering
Rochester Institute of technology | NY
Email Address: sxm4161@rit.edu

Appendix C

Recruitment Screening: Background Questionnaire

Qualtrics on line survey :

As we mentioned before the Background Questionnaire also contains the Informed Consent.

https://rit.az1.qualtrics.com/jfe/form/SV_2bIXZ6UHZEhWD7T

Accessible UML Class Diagrams Study - Background Questionnaire / Screener

Accessible UML Class Diagrams Study - Background Questionnaire

Screener

. Please complete this survey in order to help us better understand your background and use of technology.

1. Enter your Email:

2. What is your Age?

Default Question Block

3. What is your Gender:

Male
Female
Other

4. Select the option that best describes your level of vision:

Fully sighted
Totally blind
Low vision (uses a screen reader; cannot read magnified text)
Moderate visual impairment (can read magnified text)

5. How long have you been programming?

No Experience

Less than one year

One to five years

Six to ten years

More than ten years

6.

Have you ever used any object oriented programming language/technique?

Yes

No

Not sure what is object oriented programming

7. Are you currently enrolled as a student?

Not enrolled in school

Undergraduate

Graduate

8. Have you studied (or are you studying) computer science or a related field? if yes mention what it is.

9. Have you ever needed UML class diagrams for a class/project?

Yes

No

Not sure what is UML diagram.

Others, please specify

10. How do you create UML class diagrams? Mention any tools/Strategies that you use.

11. How do you read existing UML class diagrams? Mention any tools/Strategies that you use.

12.
Specify the operating system that you use most often when you program:

Mac

Linux

Windows

Other, please specify:

13. What browser do you use most often?

Google Chrome

Internet Explorer

Mozilla Firefox

Other, please specify:

14. Do you use any assistive technology when using computers? Select all that applies:

screen reader

screen magnifiers

Other, please specify:

15.

If you use screen readers, what screen reader do you use? Select all that applies:

JAWS

NVDA

Windows-eyes

ChromeVox

VoiceOver

Other, please specify:

Block 2

Q19.

Informed Consent

INTRODUCTION

Thank you for volunteering to be a part of this research study. In this research study, we are evaluating programmer use of prototype and its suggested representation of UML class diagram.

WHAT IS INVOLVED IN THE STUDY?

This study will take 60 minutes to complete. You will be asked to perform tasks to evaluate a prototype that contains multiple class diagrams and will be asked to answer questions related to the tasks performed. Your screen will be recorded while performing the tasks and notes will be taken to record your opinions and actions. The information in this study may be used with others to improve products and for educational purposes. The investigators may stop the study or take you out of the study at any time if they judge it is in your best interest. They may also remove you from the study for various other reasons. This can be done without your consent. You can stop participating at any time without loss of benefits.

RISKS

Risks associated with participating in this study are minimal. The risks involved are no greater than those involved with using any website.

BENEFITS TO TAKING PART IN THE STUDY

There are no anticipated direct benefits for participating in this study. The study will be used to help direct future research into creating more accessible UML diagrams and to inform design of potential tool for this reason.

CONFIDENTIALITY

Your name will not be used when data from this study are published. Every effort will be made to keep your research records and other personal information confidential. We will hold as confidential your personal information (such as name and phone number) and use it only for data analysis purposes, to link data to the subject. The only connection between your participation in this study and the study itself will be the signed consent form. You will be assigned a participant number. Only the participant number will be recorded on the test instruments. No personally identifiable information will be recorded on the test instruments nor stored within the software you use today. Participant identities will not be made part of any published findings resulting from this study.

INCENTIVES

At the conclusion of the study, you will enter a raffle to win $50.00 gift card.

YOUR RIGHTS AS A RESEARCH PARTICIPANT

Participation in this study is voluntary. You have the right not to participate at all or to leave the study at any time. Deciding not to participate or choosing to leave the study will not result in any penalty or loss of benefits to which you are entitled, and it will not harm your relationship with RIT.

CONTACT

For further questions , you may contact the researcher, Silva Matti at 5852841100 or via e-mail at silva.hm88 @gmail.com.

Or contact Heather Foti, Associate Director of Human Subjects Research Office at RIT at (585) 475-7673 or via e-mail at hmfsrs@rit.edu

Consent of Subject (or Legally Authorized Representative)

I did read the consent form,and By selecting the following option I'm consenting to participate in the study

[ ] Other, please indicate if you have any reasons not to participate:

Appendix D

Post Study Survey

Post Study Survey - Accessible UML Class Diagrams Study

Default Question Block

Please answer the following questions about your experience with exploring class diagrams on the web.

For each of the following statements, select your level of agreement on the scale that ranges from Strongly Agree to Strongly Disagree

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
Overall, using the system was intuitive	○	○	○	○	○
Finding different components of the class diagram was easy	○	○	○	○	○
Using the keyboard for navigation to find a specific component was helpful	○	○	○	○	○

What did you like about the system?

What would you like to see improved about the proposed approach to accessing the class diagrams? Why?

Do you have any suggestions that you would like to add?

Any other comments.

Bibliography

Albusays, Khaled and Stephanie Ludi (2016). "Eliciting Programming Challenges Faced by Developers with Visual Impairments: Exploratory Study". In: *Proceedings of the 9th International Workshop on Cooperative and Human Aspects of Software Engineering*. CHASE '16. Austin, Texas: ACM, pp. 82–85. ISBN: 978-1-4503-4155-4. DOI: 10.1145/2897586.2897616. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2897586.2897616.

Balik, Suzanne P. et al. (2013). "GSK: Universally Accessible Graph Sketching". In: *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*. SIGCSE '13. Denver, Colorado, USA: ACM, pp. 221–226. ISBN: 978-1-4503-1868-6. DOI: 10.1145/2445196.2445266. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2445196.2445266.

Balik, Suzanne P. et al. (2014). "Including Blind People in Computing Through Access to Graphs". In: *Proceedings of the 16th International ACM SIGACCESS Conference on Computers & Accessibility*. ASSETS '14. Rochester, New York, USA: ACM, pp. 91–98. ISBN: 978-1-4503-2720-6. DOI: 10.1145/2661334.2661364. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2661334.2661364.

Bardot, Sandra, Marcos Serrano, and Christophe Jouffrais (2016). "From Tactile to Virtual: Using a Smartwatch to Improve Spatial Map Exploration for Visually Impaired Users". In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*. MobileHCI '16. Florence, Italy: ACM, pp. 100–111. ISBN: 978-1-4503-4408-1. DOI: 10.1145/2935334.2935342. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2935334.2935342.

Browser Statistics. URL: https://www.w3schools.com/Browsers/default.asp (visited on 07/17/2017).

Downs, Roger M and David Stea (1973). *Image and environment: Cognitive mapping and spatial behavior*. Transaction Publishers.

Fakhroutdinov, Kirill. *UML-Diagrams | UML Multiplicity and Collections - defining and using multiplicity and collections in UML - lower and upper bounds, cardinality, order, unique.* URL: http://www.uml-diagrams.org/multiplicity.html?context=class-diagrams (visited on 07/02/2017).

Ferres, Leo et al. (2013). "Evaluating a Tool for Improving Accessibility to Charts and Graphs". In: *ACM Trans. Comput.-Hum. Interact.* 20.5, 28:1–28:32. ISSN: 1073-0516. DOI: 10.1145/2533682.2533683. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2533682.2533683.

Goncu, Cagatay and Kim Marriott (2015). "Creating eBooks with Accessible Graphics Content". In: *Proceedings of the 2015 ACM Symposium on Document Engineering*. DocEng '15. Lausanne, Switzerland: ACM, pp. 89–92. ISBN: 978-1-4503-3307-8. DOI: 10.1145/2682571.2797076. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/2682571.2797076.

Goncu, Cagatay et al. (2015). "Accessible On-Line Floor Plans". In: *Proceedings of the 24th International Conference on World Wide Web*. WWW '15. Florence, Italy: International World Wide Web Conferences Steering Committee, pp. 388–398. ISBN: 978-1-4503-3469-3. DOI: 10.1145/2736277.2741660. URL: https://doi-org.ezproxy.rit.edu/10.1145/2736277.2741660.

IBM UML basics: The class diagram. URL: https://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/ (visited on 07/02/2017).

Ishihara, Tatsuya et al. (2006). "Analyzing Visual Layout for a Non-visual Presentation-document Interface". In: *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility*. Assets '06. Portland, Oregon, USA: ACM, pp. 165–172. ISBN: 1-59593-290-9. DOI: 10.1145/1168987.1169016. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/1168987.1169016.

Kamel, Hesham M. and James A. Landay (2002). "Sketching Images Eyes-free: A Grid-based Dynamic Drawing Tool for the Blind". In: *Proceedings of the Fifth International ACM Conference on Assistive Technologies*. Assets '02. Edinburgh, Scotland:

ACM, pp. 33–40. ISBN: 1-58113-464-9. DOI: 10.1145/638249.638258. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/638249.638258.

Kaplan, Stephen (1973). "Cognitive maps in perception and thought". In: *Image and environment*, pp. 63–78.

Kennel, Andrea R. (1996). "Audiograf: A Diagram-reader for the Blind". In: *Proceedings of the Second Annual ACM Conference on Assistive Technologies*. Assets '96. Vancouver, British Columbia, Canada: ACM, pp. 51–56. ISBN: 0-89791-776-6. DOI: 10.1145/228347.228357. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/228347.228357.

King, Alasdair et al. (2004). "Presenting UML software engineering diagrams to blind people". In: *International Conference on Computers for Handicapped Persons*. Springer, pp. 522–529.

Kitchin, Robert M (1994). "Cognitive maps: What are they and why study them?" In: *Journal of environmental psychology* 14.1, pp. 1–19.

Lucidchart | Class Diagram for Online Shopping System (UML). URL: https://www.lucidchart.com/pages/class-diagram-for-online-shopping-system-UML (visited on 07/02/2017).

Lucidchart | Class Diagram Tutorial. URL: https://www.lucidchart.com/pages/uml/class-diagram (visited on 07/02/2017).

Müller, Karin (2012). "How to make unified modeling language diagrams accessible for blind students". In: *Computers Helping People with Special Needs*, pp. 186–190.

OMG UML 2.5. URL: http://www.omg.org/spec/UML/2.5/PDF/ (visited on 07/02/2017).

Papadopoulos, Konstantinos, Marialena Barouti, and Eleni Koustriava (2016). "The improvement of cognitive maps of individuals with blindness through the use of an audio-tactile map". In: *International Conference on Universal Access in Human-Computer Interaction*. Springer International Publishing, pp. 72–80.

Petrie, Helen et al. (2006). "Remote Usability Evaluations With Disabled People". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '06. Montréal, Québec, Canada: ACM, pp. 1133–1141. ISBN: 1-59593-372-7. DOI: 10.1145/1124772.1124942. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/1124772.1124942.

PlantUML. URL: http://plantuml.com/ (visited on 07/02/2017).

Péruch, Patrick et al. (2006). "Comparing distances in mental images constructed from visual experience or verbal descriptions: the impact of survey versus route perspective". English. In: *Quarterly journal of experimental psychology (2006)* 59.11, p. 1950.

Qualtrics. URL: https://www.qualtrics.com/ (visited on 07/02/2017).

Rubin, Jeffrey and Dana Chisnell (2008). *Handbook of Usability Testing : How to Plan, Design, and Conduct Effective Tests.* Vol. 2nd ed. Wiley. ISBN: 9780470185483. URL: http://search.ebscohost.com.ezproxy.rit.edu/login.aspx?direct=true&db=nlebk&AN=233103&site=ehost-live.

Section508.gov. URL: https://www.section508.gov/ (visited on 07/01/2017).

Tolman, Edward C. (1948). "Cognitive maps in rats and men." In: *Psychological Review* 55.4, pp. 189 –208. ISSN: 0033-295X. URL: http://search.ebscohost.com.ezproxy.rit.edu/login.aspx?direct=true&db=pdh&AN=1949-00103-001&site=ehost-live.

WAI-ARIA. URL: https://www.w3.org/TR/wai-aria-1.1/ (visited on 07/02/2017).

Walker, Bruce N. et al. (2013). "Spearcons (Speech-Based Earcons) Improve Navigation Performance in Advanced Auditory Menus". In: *Human Factors* 55.1. PMID: 23516800, pp. 157–182. DOI: 10.1177/0018720812450587. eprint: http://dx.doi.org/10.1177/0018720812450587. URL: http://dx.doi.org/10.1177/0018720812450587.

Which web browsers are most accessible? | DO-IT. URL: http://www.washington.edu/doit/which-web-browsers-are-most-accessible (visited on 07/17/2017).

Yoshida, Tsubasa et al. (2011). "EdgeSonic: Image Feature Sonification for the Visually Impaired". In: *Proceedings of the 2Nd Augmented Human International Conference*. AH '11. Tokyo, Japan: ACM, 11:1–11:4. ISBN: 978-1-4503-0426-9. DOI: 10.1145/1959826.1959837. URL: http://doi.acm.org.ezproxy.rit.edu/10.1145/1959826.1959837.