Master of Science in Computer Science
January 2023

Implementation and Evaluation of Software Architecture Visualization

NAGA PREETHIKA MULE

Contact Information:
Author(s):
NAGA PREETHIKA MULE
E-mail: naml20@student.bth.se

University advisor: Valeria Garro
E-mail: valeria.garro@bth.se
Department of Computer Science DIDA Department

Faculty of Computing
Blekinge Institute of Technology
SE–371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

Abstract

Background : Software architectures are often large documents containing detailed information about different architectural components, how they interact, the environment in which they operate and how the components are designed. With such a large amount of data, the sizes of the documents often get larger and it becomes challenging for humans to process in a reasonable time. Visualization is the concept of presenting data in a certain way like a graphical or pictorial format, to make data easier to understand for many people. The concept of visualization is introduced into the software architecture domain to ensure that the lengthy architectural documents are simple to understand.

Objectives : The objective of this thesis is to understand the data and relationships in the data and find an accurate visualization technique suitable for the data and implement the visualization as an interactive online tool and evaluate the visualization tool through a comparative user study with an implemented tool against existing practice.

Methods : The research method used in this thesis is Experiment, where the data for the experiment is collected through a comparative user study and collected data is evaluated using subjective and objective metrics. The objective metrics used to evaluate are 'time' and 'correctness' and the subjective metric is the NASA-TLX assessment form, which the participants are asked to fill out during the user study. The statistic value used to test the hypothesis is calculated using Wilcoxon signed rank test.

Results : The results of the experiment are such that the time spent by the users answering the questions in the implemented tool is significantly less compared to the existing method. There is not much difference in the correctness of understanding the software architecture, and the subjective workload experienced by the participants during the user study is significantly less in the implemented tool in compared to the existing method.

Conclusions : The purpose of this thesis is to create an interactive visualization tool that can effectively display the data and relationships within the data. The tool was evaluated through a comparative user study, using both subjective and objective metrics, which showed significant advantages of using it in terms of time and workload reduction.

Keywords : Software architecture, Software architecture visualization, Visualization techniques, Graph-based Visualization, NASA-TLX assessment

Acknowledgments

I would like to thank my supervisor Valeria Garro for motivating and constantly helping me throughout my thesis and encouraging in my tough times and guiding me to complete this thesis.
I would also like to thank Erik Persson X and Pär Karlsson A for giving me the opportunity to do my thesis with ERICSSON and for the constant supervision and support throughout the thesis.
Lastly I would like to thank my parents and friends for their constant support and encouragement.

Contents

List of Figures

Chapter 1
Introduction

When thinking of architecture, the immediate thought is the process of planning, designing and constructing buildings or other structures. The correct definition of architecture is the art or science of construction. In a similar manner, Software engineering [7] uses a term called **Software architecture** which refers to the fundamental structures of a software system [8] as well as the discipline of developing such structures and systems. Just creating a software architecture is not enough, A software architecture must be documented and communicated well in order for its stakeholders to use it effectively in their jobs [9].

Visualization is a method of presenting facts or concepts in a graphical or pictorial format to make them easier to understand [10]. In general, when information is gained through vision, a greater level of understanding is achieved compared to the combination of all other senses [11]. When presented with a large volume of data, it becomes challenging to quickly comprehend such a large volume of information. As a result, when presented with material in a certain way, information becomes easier to grasp. For example, bar graphs or pie charts are used to convey numerical data instead of paragraphs of text. Often times when people do not know what questions they want to ask ahead of time, they employ visualization to interpret data.

Software architectures are frequently large documents because typical enterprise software consists of a large number of components and relationships among them and understanding this may be time-consuming and at times challenging. Visualization of software architecture addresses this problem by providing data in visual formats such as images and diagrams, which improves communication and comprehension [12]. The area of software visualization focuses on using visual representations to make the software more understandable. Often various visual attributes in visualization can be styled to show interesting aspects of data [13].

This thesis will focus on visualizing the software architecture. The software architecture data used in this thesis is provided by an external company Ericsson. The discussions throughout this thesis will be based on which visualization technique to use with the data provided by Ericsson and the motivation for using the visualization technique in Ericsson and the evaluation of the implemented visualization tool.

This thesis is done together with another student from ‘Master of Science in Software Engineering’. We have worked together in Ericsson on the implementation of the visualization tool and the evaluation of the visualization tool is performed separately using different evaluation techniques and on different participants.

1.1 Software Architecture

To express it in simple terms, software architecture is the organization of a system [8]. This organization includes all components, how they interact with one another, the environment in which they work, and the software design principles. In many circumstances, it might also involve the software's future evolution.

The field of a software architecture includes a concept referred to as Architecture patterns. Architectural patterns is defining the fundamental structure of an application and can be thought of as a reusable approach to a commonly occurring problem in software architecture in a specific scenario [14].
The architectural pattern used in the context of this thesis is 'Microservice architecture' which is a subcategory of service-oriented architecture which is one of the many architectural patterns.

A microservice architecture is created as a collection of finely-grained, loosely coupled services that communicate with one another using simple protocols. A large project uses microservice architecture through small, mono-functional modules (microservices). The building blocks are created independently and combined to form the overall product [15].

In a microservice architecture, **service** is a software object that is self-contained, portable, replaceable, and reusable with a well-defined set of functionality that encapsulates its implementation and exports it as a higher-level interface that can interact with other components [15]. A separation or boundary between two distinct services that allow them to engage or communicate with one another is known as an **interface** [15]. Eventually, all of these microservices interact with each other through interfaces and create a large software system. The interaction between the services is shown in Figure 1.1

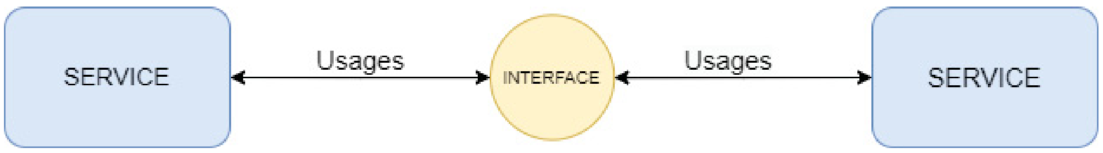


Figure 1.1: Microservice Architecture

Even the best architecture will be practically rendered useless if the people who need to use it don't know what it is about and can not comprehend it well enough to utilize it, or, worst of all, misunderstand it and use it incorrectly. All of the architecture team's analysis, hard work, and insightful design will be a waste of resources. For these reasons, all architecture should be thoroughly documented in a language that anyone can understand.

Architecture documentation serves various purposes, in fact, it helps the architects themselves as the architecture is under process [9]. As the complexity of the architecture increases, it can be time-consuming to read and understand the architecture, for this reason, we are introducing the concept of visualization into the software architecture domain.

1.2 Background

Often software systems are complex, abstract and difficult to observe. Experts must use visualization to understand the systems [13]. These issues worsen in large-scale software systems, where the system's size makes it more difficult for experts to understand their behavior and attributes. One of the critical aspects of software visualization is the ability to efficiently visualize and explore a system's software architecture [16, 17].

This thesis is done in close collaboration with an external company Ericsson. This work is done for the architecture team within Ericsson. The data in this thesis is the software architecture documentation, which includes a description of the Ericsson architectural framework.

The software architecture documentation consists of multiple services and how each service is built. Each service has multiple diagrams explaining the structure of the service and its connections. All the services are internally categorized into subcategories. This architecture is meant to capture and communicate the system's major architectural decisions. Using a variety of architectural views, the architectural data describes a comprehensive architectural overview of the system.

There are different users of software architecture documentation because different people have different roles and responsibilities within an organization. Additionally, different users may have different levels of expertise or experience with the architecture, which can also impact their use. The different users for this architecture are:

1. Designers/Developers
2. System Managers
3. Product Managers
4. Integration and Verification Personnel

Another important concept in the thesis is Unified Modeling Language, which is a graphical language used to represent large systems in a graphical diagram. Using UML diagrams for visual representation is a recognized industry standard practice. Various diagrams are used to represent various aspects of a software system. In this thesis, a UML component diagram is used. A UML component diagram is used to represent the structural relationships between different components in a large software system.

1.3 Problem Definition

This section will briefly explore the current situation at Ericsson and the issues that are the foundation of this thesis.

For a user of the architecture, who does not have much information about the software architecture of the EDA team, it is difficult to navigate the documentation between the different services and find the data they are looking for in a reasonable time. There are 47 services and 77 interfaces in the whole architecture. This number can change as the team is continuously working on improving the architecture. In the future, they can add additional services or interfaces according to the necessity.

The development of this visualization concept for the architecture was driven by the need to address the absence of documentation or a comprehensive visual representation of the interactions between the various services in the architecture at a high level of abstraction.

Another main reason was that often there is a requirement to add a new service or new interface in the existing system. Before adding a new component, the architect needs to clearly understand the dependencies of all the components have with each other and which all components will be affected when a change is made in the architecture. When there is no image or description that represents this data and it becomes difficult to understand the whole structure.

To resolve the issues specified above, the concept of visualization is introduced into this existing system. The dynamic visualization tool is developed such that users can understand how all the services are interrelated and also enable searching and filtering options for better navigation into the architecture.

1.4 Aim and Objectives

The main goal of this thesis is to design, implement and evaluate an interactive visualization tool for Software Architecture Documentation (SAD).

The objectives are:

1. Conduct preliminary interviews with the stakeholders of SAD so that there is a clear understanding of the requirements and expectations of the stakeholders regarding how the visualization tool needs to be.

2. Analyze and understand the software architecture and relationship among the components such that a reasonable visualization technique is selected to implement.

3. Implement the chosen visualization technique as an interactive online visualization tool.

4. The visualization tool should be implemented such that it should have functionalities of supporting searches, filtering options and navigating between different parts of the SAD.

5. Evaluate the implemented tool using Subjective and Objective Metrics.

1.5 Research Questions

The following research questions are formulated to address the concerns raised above, and a more detailed description is provided in Chapter 3.

Research Question 1:

Does the use of the implemented visualization tool reduce the time taken to understand the structure of a software architecture compared to using documentation and

a UML component diagram?
Justification : Measuring the time a user spends on a tool is a crucial aspect in evaluating its performance and effectiveness. This question is designed to provide a quantitative measurement of a user's behavior, allowing for an objective comparison between the tool and the documentation with UML component diagram. The answer to this question will provide valuable insights into the relative performance of the implemented tool and help determine if it is performing better than the documentation and UML component diagram.

Research Question 2:
Does the use of the implemented visualization tool improve the correctness of understanding the structure of software architecture and answering architectural questions as compared to using documentation and UML component diagram?
Justification : The formulation of this question is based on the importance of determining if users have a clear understanding of the architecture and are able to effectively find the information they need using the implemented visualization tool. Knowing this information is crucial in determining if users will prefer using the tool over other methods, such as the documentation and UML component diagram.

Research Question 3:
Does using the implemented visualization tool reduce the user's subjective workload when working with the software architecture compared to using the documentation and UML component diagram?
Justification : As one of the primary objectives of using visualization is to decrease workload, it is essential to determine if the implemented visualization tool is indeed reducing the subjective workload experienced by the user. Measuring subjective workload can provide valuable insight into the effectiveness of the tool in meeting this goal.

1.6 Ethical, Societal and Sustainability Aspects

In this thesis work, the following Ethical aspects have been considered:

1. The privacy and confidentiality of the company's architecture will be maintained. Data anonymity is maintained throughout the research because of the presence of the external company.

2. Since in this thesis, for evaluation we conduct a user study. he anonymity of participants and the anonymity of the information provided by the participants is maintained.

3. The participants are made aware that the user study is completely voluntary and the participants are free to discontinue at any time in between the user study without giving an explanation.

1.7 Outline

Chapter 1 of this thesis paper begins with the introduction of the concepts and provides a background on various topics, problem formulation, aims, objectives and research questions are discussed. Chapter 2 summarizes most relevant previous research in this area that will serve as a solid foundation for this thesis. The research methodology, research design and evaluation process are covered in Chapter 3. The results and analysis of the thesis are presented in Chapter 4. Chapter 5 contains discussions regarding the research questions and thesis as a whole and discusses threats to validity and chapter 6 contains conclusion and future work.

Chapter 2

Related Work

Software architecture visualization has emerged as an essential aspect of software engineering over the past two decades. The software community has developed many methods and tools to help understand high-level architecture design. However, there are some differences that distinguish this work from others.

This section will include a description of previous studies on software architecture visualization, visualization tools, and evaluation methods for software architecture visualization and visualization tools, including how and in what domains these tools are implemented, among other things.

In [12], the systematic literature review is performed on software architecture visualization techniques, and 53 papers were selected based on pre-defined inclusion and exclusion criteria. This study aimed to investigate and identify the various types of visualization techniques used, what is the relationship between different visualization techniques and architecture activities and what are the different purposes of using visualization techniques. This paper shows that the most popular visualization technique is Graph-based visualization among Notation-based visualization, Matrix-based visualization, and Metaphor-based visualization. These different categories are further explained in section 2.1. This paper presents evidence that graph-based visualization is a widely used technique in architectural activities. This paper also has ten categories in a section explaining the purpose of using visualization techniques such as Improve the understanding of architecture evolution, Improve the understanding of static characteristics of architecture, Improve search, navigation, and exploration of architecture design. This paper also states that the main method for evaluating architecture visualization is the industrial study, commonly done as a case study. The data in this thesis supports the use of a graph-based technique, As a result, the implementation of the visualization tool in this study adopts a graph-based technique. Another concept relevant to this thesis from this paper is the purpose of using visualization techniques; the category relevant to us in this study is improving search, navigation, and exploration of architectural design. This thesis is entirely based on the idea of improving search in our complex software architecture.

Software systems are often large, complex systems consisting of thousands of hierarchically organized elements like directories, sub-directories, files, and functions. Often these hierarchy elements can carry additional information worth investigating for a software developer. Burch et al. in [18], have implemented hierarchy visualization based on the visual metaphor of indentation to generate an overview of the software system hierarchy and have a functionality to easily attach additional attributes. They have also provided interactive functionality like filtering, hierarchy transformations, and details on demand. This paper serves as a reference for the current thesis, as it presents a user study that compares the effectiveness of indented hierarchy visualization against the traditional node-link diagram using datasets of three sizes. This paper emphasized how important it is to have an overview of a system, which is one of our motivations for this thesis.

In [19] authors developed an interactive visualization using a conversational user interface rather than a graphical user interface. The visualization technique used in this paper is graph visualization. This visualization gets events from the chat-bot, filters them, modifies the graphic accordingly and finally, it is integrated with previously developed visualization of OSGi-based projects [20]. Similarly, [18] also highlights the challenge to preserve an overview of the software architectures with the modern software systems and their complexities.

To gain more understanding of the software maintenance activities, EXTRAVIS tool was used. EXTRAVIS is a tool for the visualization of large traces. The first controlled experiment was carried out by Cornelissen et al. in [21,22] which provides a quantitative evaluation of trace visualization for program comprehension. It also identifies the tasks for which EXTRAVIS is most helpful and provides empirical evidence that the additional availability of this trace visualization tool can provide benefits with regard to time and correctness.

As an extension to this, Fittkau et al. [23]performed two controlled experiments with different-sized object systems to compare the trace visualization tools ExtraVis and ExplorViz in typical program comprehension tasks. ExplorViz provides multi-level visualization from the software landscape layer toward the level of individual software applications [24]. The participants used both tools for a small-sized system equally to perform program comprehension tasks, employing ExplorViz resulted in a significant efficiency edge of 28 percent less time spent by using ExplorViz. ExplorViz significantly increased correctness for both object system sizes with 39 and 61 percent in terms of effectiveness.

In [25] Lam et al. have formulated evaluation scenarios of information visualization, which they categorized into two main subcategories which are the scenarios for understanding the data analysis process and the scenarios for understanding visualizations. These are later divided into seven scenarios understanding environments and work practices (UWP), evaluating visual data analysis and reasoning (VDAR), evaluating communication through visualization(CTV) evaluating user performance (UP), evaluating user experience (UE), and evaluating visualization algorithms (VA). This paper provides an overview of the seven scenarios and makes it clear for the researchers and practitioners to understand and choose the right evaluation goals. Each evaluation scenario describes the evaluation goal and outputs evaluation questions and common approaches in that scenario. After understanding the different evaluation scenarios, the evaluation method User Performance is selected for this thesis.

The research [26] is an Empirical evaluation of Linked Data visualization tools, in which they conducted a function-based evaluation and comparison of different tools. Generally Linked Data visualization aims to provide graphical representations of datasets. This paper provides an overview of previous graph visualization research as well as some inspiration for how a graph visualization tool should be implemented. Some of the graph-based visualization techniques discussed in this paper include

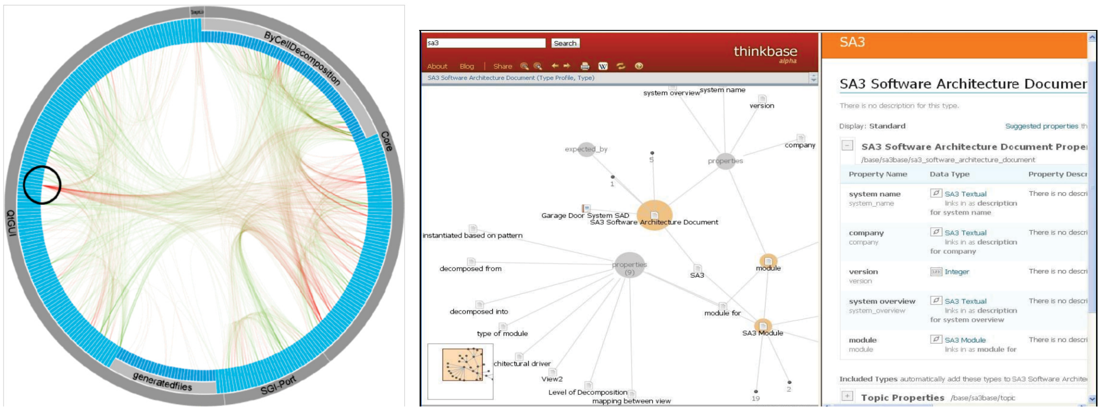
Aemoo [27], Graphless [28], H-BOLD [29], graphvizDB [30].

2.1 Visualization Techniques

The various visualization techniques that are currently used in software architecture over time are Graph-based visualization, Notation-based visualization, Matrix-based visualization, and Metaphor-based visualization [12].

A form of data representation approach known as graph-based visualization makes use of nodes and edges to visually illustrate the relationships between various components. In this paper, we adopt the use of graph-based visualization to depict the relationships between elements in the architecture. The use of graphs is motivated by the need to effectively communicate connections between elements that are considered relevant and useful. This approach is considered suitable when a comprehensive understanding of the overall structure is desired.

An example of how a graph-based visualization can be implemented is demonstrated in Figure 2.1.



SA3

SA3 Software Architecture Documer

There is no description for this type.

Display: Standard　　Suggested properties th

SA3 Software Architecture Document Prope
/base/sa3base/sa3_software_architecture_document

Property Name	Data Type	Property Descr
system name system_name	SA3 Textual links in as description for system name	There is no descri
company company	SA3 Textual links in as description for company	There is no descri
version version	Integer	There is no descri
system overview system_overview	SA3 Textual links in as description for system overview	There is no descri
module module	SA3 Module links in as module for	There is no descri

Included Types automatically add these types to SA3 Software Archite

Topic Properties /base/sa3base/topic

Figure 2.1: Graph-based Visualization (from [1, 2])

Notation-based visualization: A way of expressing information or data using symbols, forms, and icons, as well as other graphic components, that adhere to a set of rules or norms, is known as notation-based visualization. Three modeling methodologies are combined in this category: SysML (systems modeling language), UML (unified modeling language), and specific notation-based visualization. A general-purpose modeling language called SysML is used to specify, analyze, create, verify, and validate complex systems, including software, hardware, and information systems. UML is a general-purpose, industry-standard visual modeling language used to specify, design, construct, and document software-intensive systems. Mostly in notation-based visualization, custom notations are used while developing a tool. Examples of notation-based are shown in Figure 2.2.

Matrix-based visualization: Matrix-based approaches show dynamic graphs as adjacency matrices, where intra-cell timelines show the dynamics of edge and node weights, an example of a matrix based is shown in Figure 2.3.

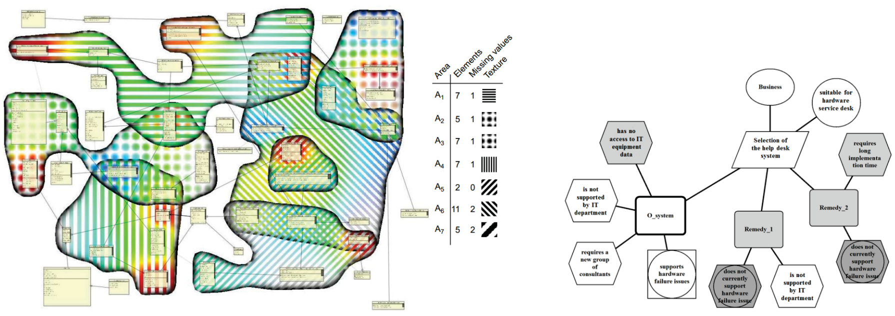

Area
Elements
Missing values
Texture
A1 7 1
A2 5 1
A3 7 1
A4 7 1
A5 2 0
A6 11 2
A7 5 2
Business
suitable for hardware service desk
has no access to IT equipment data
Selection of the help desk system
requires long implementation time
is not supported by IT department
O_system
Remedy_1
Remedy_2
requires a new group of consultants
supports hardware failure issues
does not currently support hardware failure issues
is not supported by IT department

Figure 2.2: Notation-based Visualization (from [3, 4])

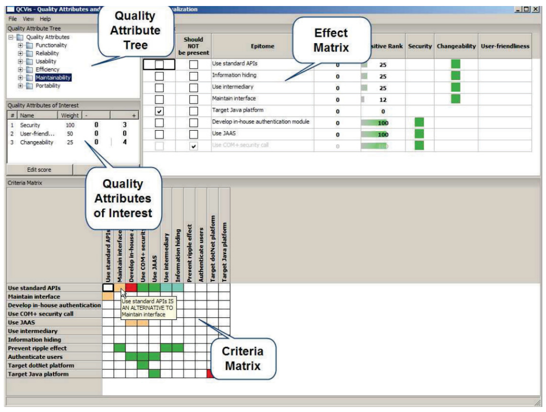


Figure 2.3: Matrix-based Visualization (from [5])

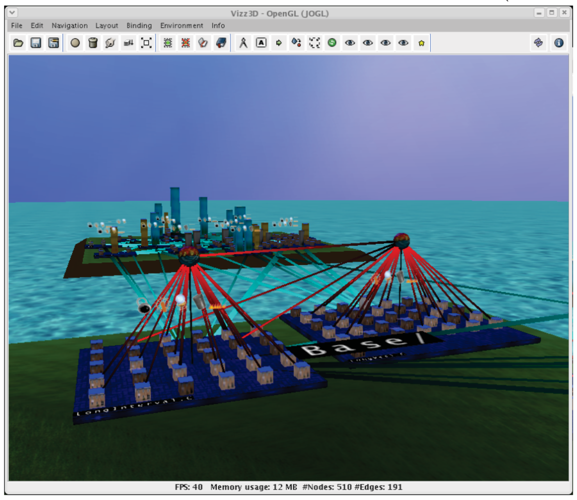

Vizz3D - OpenGL (JOGL)
File Edit Navigation Layout Binding Environment Info
FPS: 40 Memory usage: 12 MB #Nodes: 510 #Edges: 191

Figure 2.4: Metaphor-based Visualization (from [6])

Metaphor-based visualization: In the general context, a visualization metaphor is described as a map showing the relationships between ideas and things in the application domain being modeled and a set of analogies and comparisons. In this context, Metaphor-based visualization builds familiar physical world contexts (e.g., cities) to visualize software architecture entities and their relationships. Figure 2.4 depicts how a metaphor-based visualization can be.

Chapter 3

Method

This section will describe the research method used to answer the research questions. This section also includes information on the research method setup and procedure for answering the research question.

3.1 Research Method Selection

A research can be conducted in many ways for example, Experiment [31], Case study [32], Survey [33] and so on. The research method selected for this thesis is Experiment and the data for the experiment is collected through a User Study.

Experiment: A experiment is an investigation of a testable hypothesis where one or more independent variables are manipulated to measure their effect on one or more dependent variables [34]. Generally, experiments involve more than one treatment to compare the outcomes. In this thesis, the comparison is between the implemented visualization and with the documentation and UML component diagram [31].

The data gathered during the user study will be subjected to quantitative analysis. The purpose of using quantitative analysis in this thesis is driven by its focus on numerical, quantifiable data, which is most often seen as more accurate or valuable than qualitative research. Quantitative data can be generalized to larger populations if a sufficient number of participants are included in the study. On the contrary, qualitative data is often more in-depth and reliable when the focus is on individual participants [35].

User Study: By definition, user studies are about people, behavior, and contexts [36].In order to evaluate the effectiveness of the visualizations created, an attempt was made to conduct a user study with a group of employees from Ericsson.

3.2 Research Design

The research for this thesis starts with collecting requirements and data from Ericsson. This is an important step as the understanding of the requirements and data is critical to the process. The subsequent step involves identifying a challenge within Ericsson's current setup and explaining its potential impact on the field of research, as well as proposing a solution for Ericsson. Upon a clear understanding of the problem, research questions are then formulated.

The objective of this thesis is to develop a visualization tool for software architecture documentation, with the aim of improving user experience and convenience.

To achieve this goal, it is necessary to have a comprehensive understanding of the stakeholders needs and expectations from the visualization. This understanding is obtained through conducting preliminary interviews with the stakeholders.

The next step in the research design is to carry out the implementation of the visualization tool as per the identified requirements. Subsequently, the tool is evaluated through a comparative user study, which allows for a comparison of the visualization tool with existing software architecture documentation. The results of this study are documented and presented in a comprehensive manner for further analysis and future reference.

Figure 3.1 describes the research design flow.

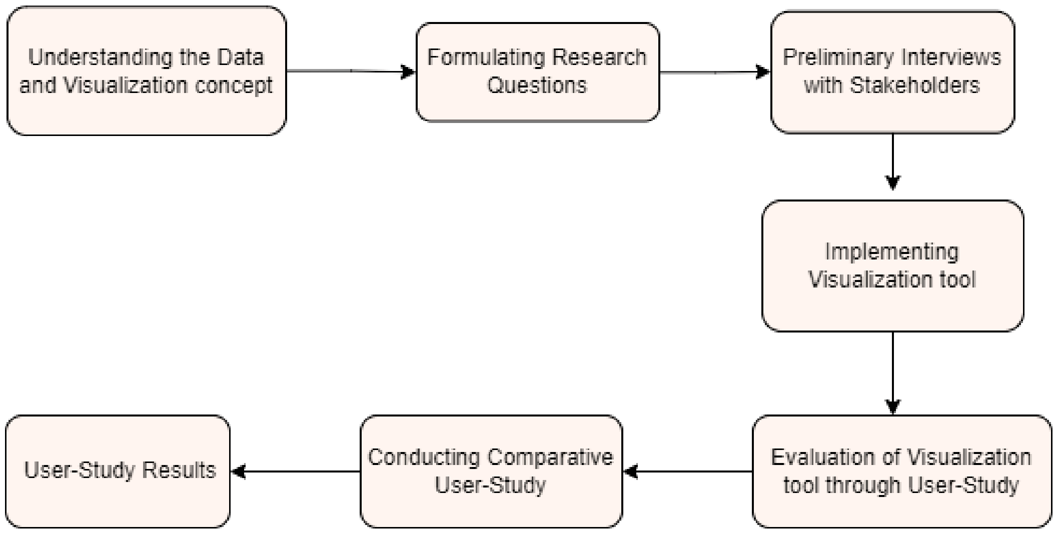

Understanding the Data and Visualization concept
Formulating Research Questions
Preliminary Interviews with Stakeholders
Implementing Visualization tool
Evaluation of Visualization tool through User-Study
Conducting Comparative User-Study
User-Study Results

Figure 3.1: Research Design

3.3 Data Collected from Ericsson

The data used in this thesis is provided by Ericsson in the form of a JSON file, which serves as the basis for the visualization of the software architecture. The use of JSON as a data format is a common choice for web applications as it is easily exchangeable, compact, text-based, human-readable, and can be edited with a simple text editor [37]. The JSON file provided by Ericsson is divided into two sections as 'services' and 'interfaces'. Each service has two relationships, 'exposed interfaces' and 'consumed interfaces' that connect to the interface section.

3.4 Preliminary Interview

The whole vision of developing a visualization is for the users to spend less time and get the most out of it. Initially, after getting the requirements from the company, the next step was to conduct a preliminary interview to collect the user goals and user personas from the different stakeholders. Understanding the target audience is fundamental to creating good visualization. Understanding the user persons helps the understanding of the expectations and concerns of users and what they are looking for in architecture. This will help to design a visualization that will satisfy the user's needs. Exploratory pilot interviews are conducted to collect information.

Interviews are generally categorized as structured, unstructured and semi-structured [31]. In this thesis, semi-structured interviews are conducted, which means that the questions in the user study can be asked in a different order, and participants can give additional information during the user study according to the context if desired.

The interviews conducted were recorded by taking proper consent from the stakeholders. The interviews are recorded to serve as a reference for future reference if needed.

The questions asked in this preliminary interview are :

1. What is your role in the organization?
2. How frequently do you refer to the software architecture?
3. How familiar are you with the architecture?
4. What are your use cases with the architecture?
5. What are your goals when exploring architecture?
6. What information do you need to achieve this goal?
7. How much time do you spend looking or browsing the architecture before you are able to act on their goals? What do you look for in this exploration?
8. How do you explore the architecture now?
9. How would you expect to be able to explore the architecture?
10. What are the pain points? What information can't you get from the architecture? What information could you not trust? Where else do you need to look/whom else do you need to talk to?
11. What applications/services is your work mostly focused on?

3.5 Implementation of Visualization tool for Software Architecture Documentation

After conducting a thorough investigation of the requirements provided by stakeholders and analyzing the data provided by Ericsson, it was determined that the data consisted of multiple interconnected components. Upon further examination, it was concluded that the relationships present in the data would best be represented through the use of graph-based visualization techniques.

The main programming language used to implement the tool is **javascript** and the libraries used are D3.js with Nodejs, HTML, and CSS. To start with, the installation of a Node.js application with the Express library was carried out, followed by the integration of D3.js with it. Node.js is a JavaScript runtime environment for the back end. Express is the leading Node.js web server library, it nicely abstracts Node.js's ability to open ports and serve content into an easy-to-use API that is very light and fast. D3.js is a popular JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS [38].

D3.js provides a large range of graph-based visualization [39] depending on the type of relationship in the data, some of the examples are Radial Tree Diagrams, Enclosures, Force Directed, Circle Packing, Node & Edge Visual Encodings.

In this thesis, Force Directed Graph visualization was employed as the foundational concept, and the visualization was created using the data. Force-directed graph visualization is a well-known technique for displaying complex networks and relationships. It uses physics concepts to establish an attractive or repulsive force between nodes, resulting in a clean and visually appealing layout. This type of graph allows for interactivity, with the ability to make real-time modifications such as moving nodes, zooming and highlighting connections for improved exploration. Force-directed visualization is scalable and widely used across various fields to study complex systems.

The implemented visualization tool is shown in Figure 3.2.

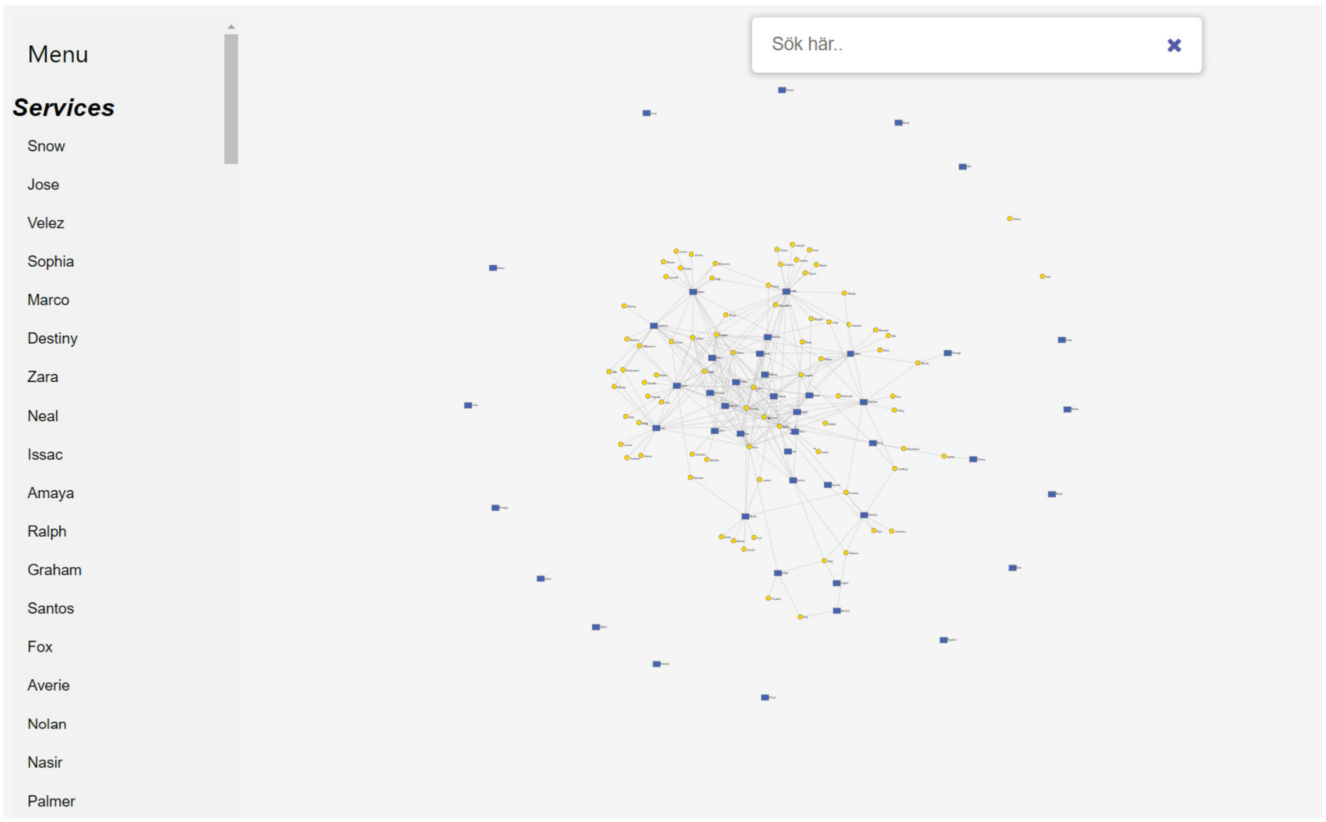


Figure 3.2: Implemented Visualization Tool Overview

Upon opening the online user interface, an interactive representation of the architecture becomes visible. The square shapes in the visualization depict services while the circular shapes represent interfaces. These services have two interface relationships as exposed interfaces and consumed interfaces. When a service has an exposed interface relationship, the arrow points toward the service. Conversely, when a service has a consumed interface relationship, the arrow points away from the service and toward the interface.

The relationship between service and interface is shown in Figure 3.3.

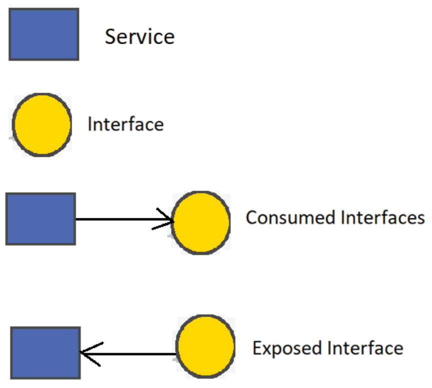


Figure 3.3: Relationship between Service and Interface

Additionally, the visualization tool includes a search feature that allows users to easily locate specific services or interfaces. When a service or interface is searched, all related connections are displayed. The search feature also includes a suggestion function that presents possible matches as the user begins typing in the search box, making it easier to find what they're looking for. Figure 3.4 illustrates the search box feature that provides suggestions.

The visualization tool also includes a zoom function, which enables the user to adjust the scale of the graph. This function is particularly useful as it provides the ability to view either a high-level overview or a more detailed look at the graph, depending on the user's needs. The zoom capability allows the user to easily concentrate on specific areas of the graph, thereby enhancing the understanding of the details present in the graph.

Figure 3.6 demonstrates that when the 'patel' interface is searched, all the connected services are highlighted in the graph, providing a clear illustration of its connections.

In Figure 3.5, when searching for a specific service, such as 'gates', the visualization tool highlights all the connected interfaces.

In the user interface, there is a convenient sidebar located on the left-hand side that lists all the names of the services and interfaces. This serves as a quick reference for users who may not have the exact name in mind to search for. By browsing through the names in the sidebar, users can easily select the desired services or interfaces. Upon clicking on a name in the sidebar, the corresponding element and all its connections will be highlighted in the graph, providing a similar experience to using the search box feature.

The selection of colors for the squares and circles in the visualization was made with care and consideration for individuals with color blindness. The two colors, blue and yellow, were chosen as they are easily distinguishable for those with color vision deficiencies [40]. This ensures that the visualization can be effectively used by a wider range of users.

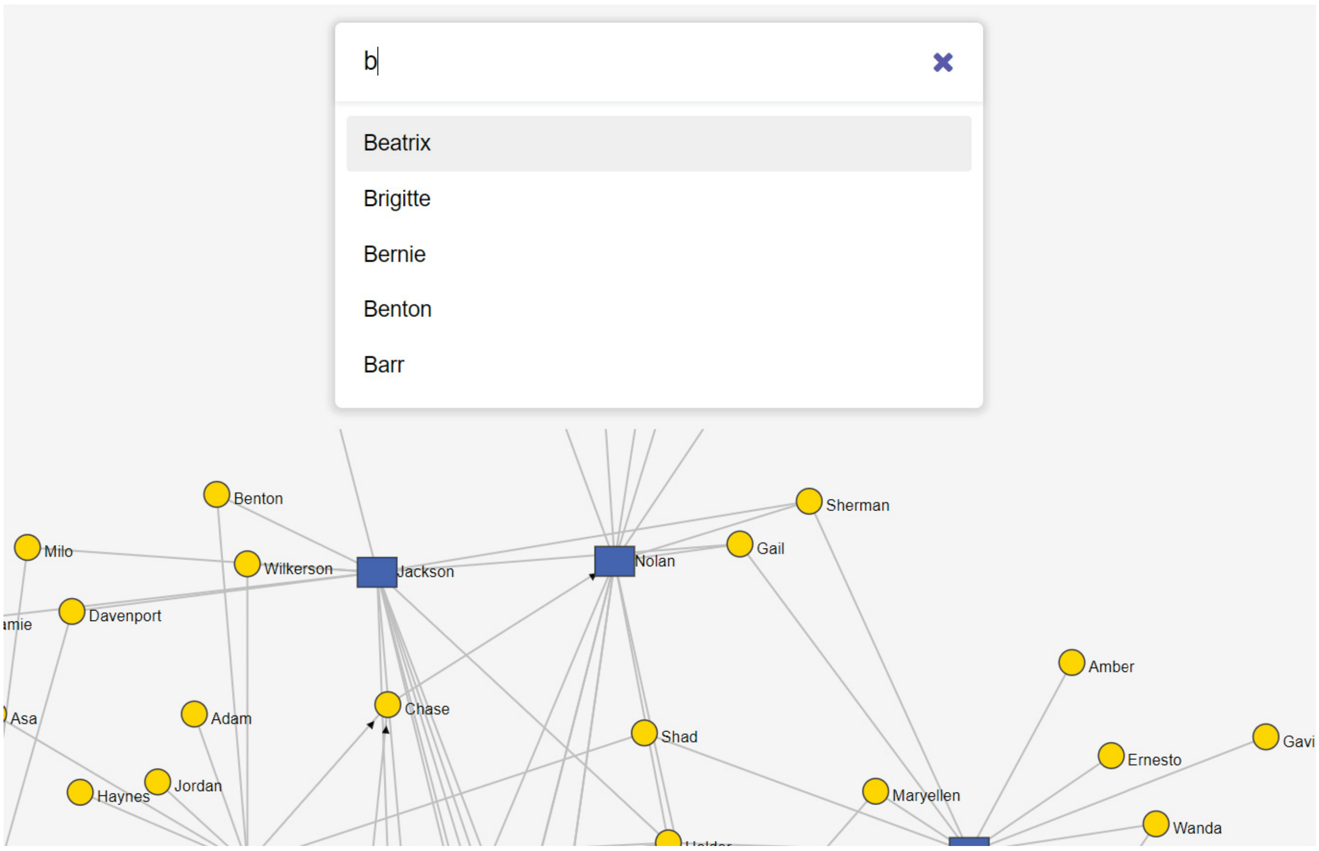


Figure 3.4: Search Box Implementation in the Tool

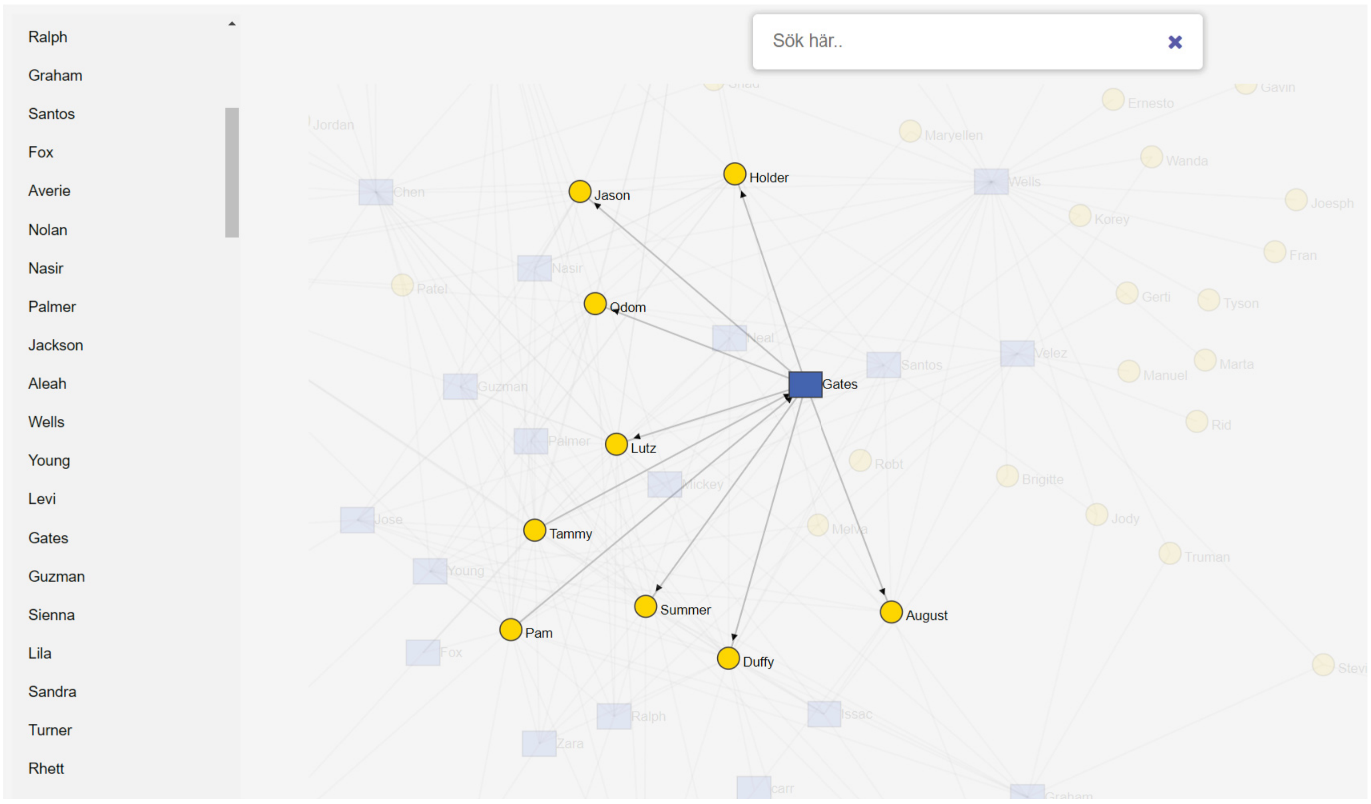


Figure 3.5: Service Highlighted

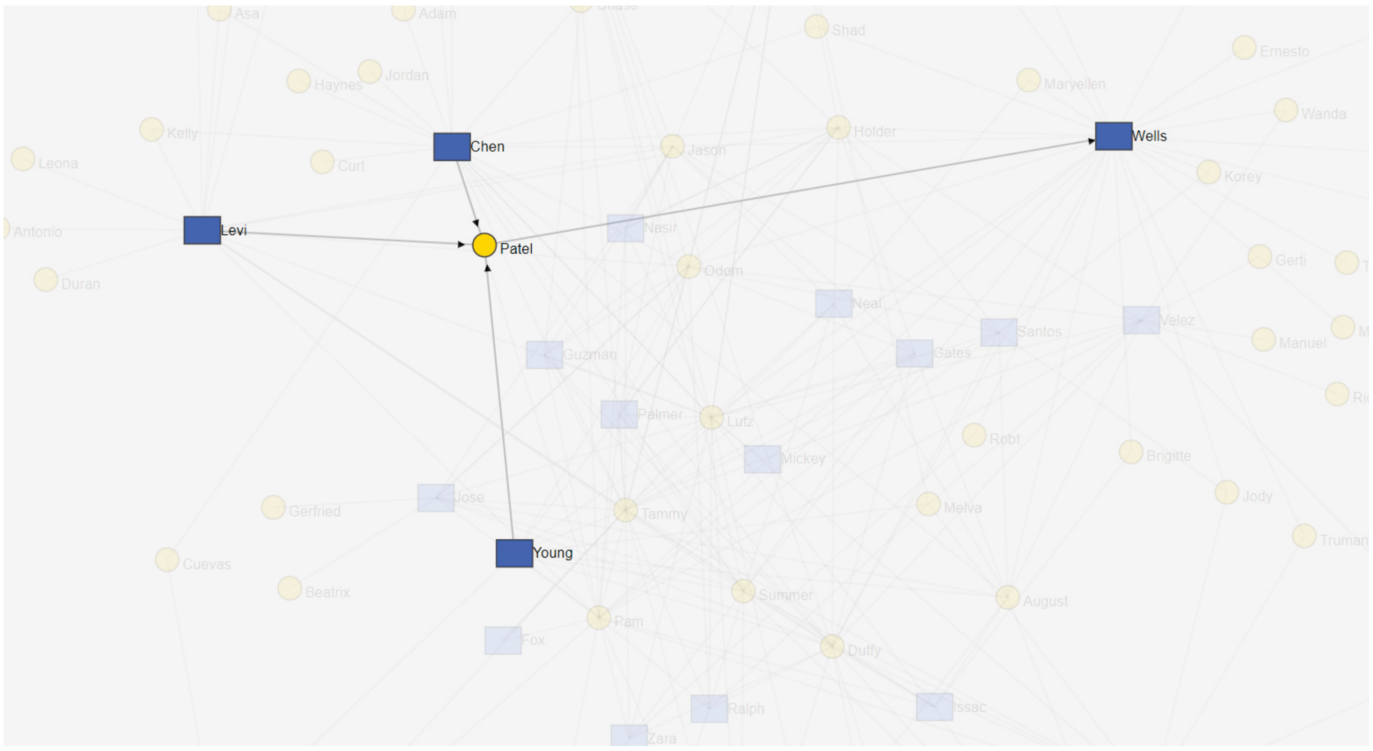


Figure 3.6: Interface Highlighted

3.6 User Study

In this study, we aim to assess the effectiveness of graph visualization tool in comparison to documentation with regard to three crucial aspects: the amount of time required to answer architectural questions, the correctness of answering the architectural questions and minimizing the cognitive load experienced by users. In line with this objective, this study aims to compare the subjective workload experienced by users when using the visualization tool and when using documentation with a UML component diagram.

As part of this study, an experiment was conducted and data for the experiment is collected through conducting a comparative user study to evaluate the three aspects as mentioned above.

Three research questions have been formulated:

RQ1: Does the use of the implemented visualization tool reduces the time taken to understand the structure of a system architecture compared to using documentation and a UML component diagram?

RQ2: Does using implemented visualization tool improved the correctness of understanding the structure of system architecture and answering architectural questions as compared to using documentation and a UML component diagram?

RQ3: Does using the implemented visualization tool reduce the user's subjective workload when working with the software architecture compared to using the documentation and UML component diagram?

Based on the Research Questions, Null Hypothesis is formulated as

$H1_0$: The use of a visualization tool does not impact the time taken to answer the

architectural questions.
$H2_0$: The use of a visualization tool does not impact the correctness of understanding the overall structure of the system architecture.
$H3_0$: The subjective workload on the users is the same while using the visualization tool and the documentation with the UML component diagram.

Alternate Hypotheses to the above Null Hypothesis for this thesis are:
H1: The use of the visualization tool reduces the time taken to answer architectural questions.
H2: The use of the visualization tool increases the correctness of the answers to architectural questions.
H3: The use of the visualization tool decreases the experienced subjective workload on the users while answering the questions during the user study.

In order to test the hypotheses $H1_0$, $H2_0$ and $H3_0$ a set of questions related to software architecture were developed and presented to participants during the user study. The procedure for conducting the user study is outlined in detail in Section 3.6.2. The statistical method chosen to test the hypotheses was the Wilcoxon Signed Rank Test, and a description of this method is provided in Section 3.6.5.

The NASA Task Load Index (NASA-TLX) assessment was utilized to assess the subjective workload of participants in this study. After answering the questions, each participant was asked to fill out the NASA-TLX form to provide a comprehensive evaluation of their subjective workload. The following section 3.6.6 provides a description of the NASA-TLX assessment.

3.6.1 Variables

According to standard practices in empirical software engineering, as outlined in [41], The dependent variables recognized are 'time', 'correctness' and 'subjective workload'. The availability of the implemented visualization tool and access to the architecture documentation serves as the independent variable.

3.6.2 User Study Procedure

This section will provide an in-depth explanation of the design of the user study and a step-by-step overview of the study procedure.

Throughout this thesis, Scenario 1 is defined as the architecture documentation and UML component diagram. On the other hand, Scenario 2 is described as the implemented visualization tool.

The steps of the user study, as depicted in Figure 3.7, are outlined below.

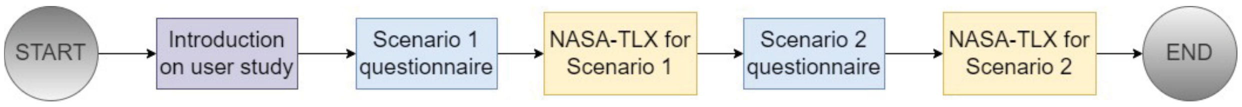


Figure 3.7: User Study Design Flow

The user study begins with briefing the participants with an introduction to what the user study will be about, the different parts of the user study, a brief introduction to the architecture and what the participants can expect from the architecture, and what the participant's task is in the user study. The user study follows all basic guidelines and obtains proper consent from participants, both verbally and through the questionnaire. The participants are informed in the introduction that they have the freedom to discontinue the study at any time without explanation. The participants are also informed that it would approximately take 25 minutes to complete the whole user study.

The next phase of the user study involves presenting two scenarios to participants and gathering their responses. In scenario 1, participants are presented with an introduction and asked to answer four architectural questions and the time taken to answer each question was recorded. After answering the questions, they are asked to fill out a NASA TLX assessment form. Similarly, in scenario 2, participants are introduced to the scenario and asked to answer four architectural questions and the time taken to answer each question was recorded. Once the participants have completed the questions, they are again asked to fill out a NASA TLX assessment form to measure their experience.

This study employs the use of two NASA TLX assessment forms in order to determine the subjective workload of two scenarios separately. Upon the completion of filling out the second assessment form, the user study is concluded and the participants are sincerely thanked for their invaluable cooperation and time invested in the study.

In this comparative user study, the focus is on comparing the implemented visualization with the existing documentation and a UML component diagram. To conduct a valid comparison, it is crucial to have two similar tools to compare. Unfortunately, Ericsson does not have any visual representation of the overall structure of its components and their relationships. To establish a proper baseline for the comparative study, a UML diagram was manually created using the StarUML tool [42], utilizing the data provided by Ericsson.

This user study was conducted remotely through a virtual meeting setup on Microsoft Teams, where participants were asked to share their screens during the session. The entire study was supervised remotely, allowing for the clarification of any questions regarding the setup or architectural questions during the course of the user study. An online questionnaire was created using a tool, where each question was equipped with a timer that automatically started when the question appeared and stopped when the participant moved on to the next question. To avoid any biases in the answers, the participants were kept unaware of the ultimate goal of the user study.

This user study was designed as a within-subject study [43], meaning that each participant answered questions in both scenarios. In this case, the same person responded to four questions related to the visualization tool and four questions related to the documentation and UML component diagram. The advantage of using a within-subject design is that it provides a strong basis for comparison, as the user's performance remains constant across both scenarios. Additionally, using this design minimizes the number of participants required for the study, making it ideal for our situation. While there are clear advantages to using a within-subject study design, it is also important to recognize its limitations. In addition to the advantages of the system, it is important to acknowledge its limitations. One of the limitations is that the user study can take a considerable amount of time to complete. Another issue is the presence of a carryover effect, which can impact the results of the study. Furthermore, when collaborating with an external company, it can become challenging to secure participants for the user study.

The user study was conducted utilizing a counterbalanced design methodology [44]. This design technique effectively manages the impact of variables in situations where participants are repeatedly exposed to different conditions. In this user study, a sample of 15 participants was used. Seven participants are randomly assigned to answer Scenario 1 followed by Scenario 2, while eight participants are asked to answer Scenario 2 prior to Scenario 1.

3.6.3 User-Study Questions

In the user study, the participants are requested to answer four architectural questions for each scenario. The description of these questions is described in this section.

In this study, the UML component diagram consists of the data provided by Ericsson, while for the purpose of this user study, the data in the implemented visualization tool is randomly generated names with the same architectural structure as the Ericsson data, but with different names. This was done to eliminate any carryover effect and ensure accurate results.

In the formulation of the questions, two critical criteria were considered: first, the questions should accurately represent actual architectural questions, and second, they should not exhibit bias towards either the implemented visualization tool or the UML component diagram-based architectural documentation.

Four questions are formulated for each scenario, but the sets of questions in the two scenarios are identical except for the name of the component they refer to. For example, in scenario 1 the component is named ‘commit’, while in scenario 2 it is referred to as ‘randal’. These names have been mapped to the same component, but in order to eliminate any carryover effects the names are changed for scenario two.

The questions asked during the user study are:

- Question 1:
  There is a service called ‘service-x’. What are the consumed interfaces exposed interfaces of this service? Please write the names of all these interfaces, that are Exposed Interfaces and the Consumed Interfaces.

- Question 2:
  What do you think are the two ‘hotspot’ services, i.e., the two services having the highest number of interfaces?

- Question 3:
  If the exposed interfaces of ‘service-x’ have to be modified, which services consuming these interfaces are affected?

- Question 4:
  Do ‘service-y’ and ‘service-z’ have consumed interface(s) in common? If yes, write the name(s) of the interface(s) and write also the name(s) of the service(s) that expose the interface(s).

To make this user study more realistic, the answers to these questions are asked to be filled in a textbox rather than multiple choice questions, making it more difficult for participants to guess and have genuine answers.

The formulation of these questions was the result of engaging in discussions with Ericsson employees, gathering insights from preliminary stakeholder interviews, and gaining knowledge about the software architecture.

3.6.4 Participants

The user study primarily involves developers from various teams within Ericsson as participants. It is important to note that the selected individuals have no prior knowledge or experience with the software architecture used in this thesis. This ensures that the results are not influenced by their prior knowledge. All participants voluntarily participated in the study, indicating their motivation towards contributing to the research.

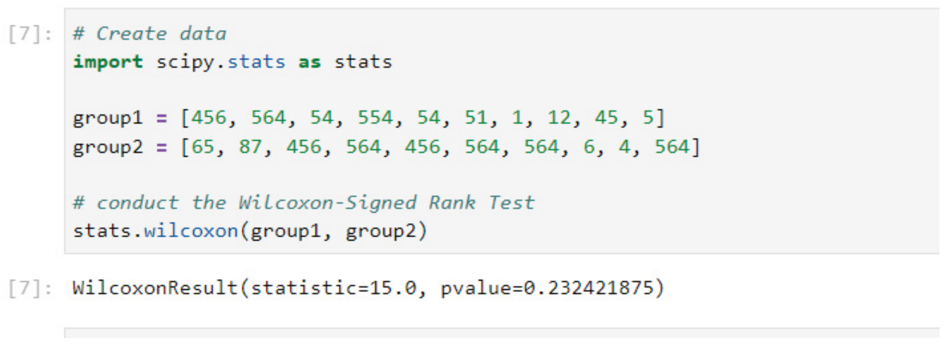
3.6.5 Wilcoxon Signed Rank Test

The Wilcoxon signed-rank test is a statistical method used to compare data from one population that has two effects. It is a non-parametric test, meaning that the data does not follow a specific pattern or distribution [45].

This test is a non-parametric alternative to the paired Student t-distribution. The Wilcoxon signed-rank test works by comparing the median of the sample data to a hypothetical median. This is done to determine whether the sample data deviates significantly from the hypothetical median, providing insights into the characteristics of the data.

The Wilcoxon signed-rank test is suitable for this user study as the sample size is small and the test is designed to compare one population with two effects.

The calculation of the statistic value for the Wilcoxon test does not require manual computation of formulas. Instead, a convenient solution is provided by the open-source library scipy [46] in the Python programming language. The scipy.stats package within scipy offers a wide range of statistical functions and probability distributions, making the Wilcoxon test simpler to perform. Data can be entered manually or loaded from a CSV or excel file in Python, as demonstrated in the Figure 3.8 using an example data set.



```
[7]: # Create data
     import scipy.stats as stats

     group1 = [456, 564, 54, 554, 54, 51, 1, 12, 45, 5]
     group2 = [65, 87, 456, 564, 456, 564, 564, 6, 4, 564]

     # conduct the Wilcoxon-Signed Rank Test
     stats.wilcoxon(group1, group2)

[7]: WilcoxonResult(statistic=15.0, pvalue=0.232421875)
```

Figure 3.8: Example Wilcoxon Signed Rank Test using Python

The Wilcoxon signed-rank method generates a test statistic value, known as a z-score, which is converted into a ‘p-value’. The p-value represents the likelihood that both populations are identical. A smaller p-value implies a more considerable difference between populations.

3.6.6 NASA TLX Assessment Test

The NASA-TLX is a multidimensional tool used to assess the mental workload of an individual while they are performing a task. The third research question is addressed by utilizing the tool. It is a popular evaluation tool created by The Human Performance Group at NASA’s Ames Research Center [47] and has been used in various domains including healthcare, aviation and other complex socio-technical domains [48, 49]. The tool requires the users to answer six subjective questions to determine their mental workload.
The six subjective questions are :

- Mental Demand:
  How mentally demanding was the task?
- Physical Demand:
  How physically demanding was the task?
- Temporal Demand:
  How hurried or rushed was the pace of the task?
- Performance:
  How successful were you in accomplishing what you were asked to do?
- Effort:
  How hard did you have to work to accomplish your level of performance?
- Frustration:
  How insecure, discouraged, irritated, stressed, and annoyed were you?

The NASA-TLX assessment tool consists of six subjective subscales that are graded using 7-point scales, ranging from very low to very high, except for the performance subscale which ranges from perfect to failure. In general, when the scores for a particular subjective question are low on the scale, it can be inferred that the users experienced a lower level of workload.

The NASA-TLX tool, being open-source in nature, enables utilization by individuals and organizations across the world without seeking prior permission or authorization from NASA. Additionally, the tool may be altered, such as through language translation, without encountering any limitations.

Chapter 4

Results and Analysis

This chapter presents the results of the research questions, and user study and provides an analysis of the collected data.

4.1 Preliminary Interview with Stakeholders

This study involved conducting interviews with employees of Ericsson. In order to maintain the confidentiality and privacy of the stakeholders as per ethical principles, the names of the stakeholders have been omitted. The stakeholders will be referred to by their positions within the organization. Five stakeholders with varying roles in the organization, including architects, managers and developers, were interviewed.

The conclusions drawn from the preliminary interviews are as follows: The architects in the team, work a lot with the architecture and are very familiar with the architecture. They use many different kinds of information and are the ones who know architecture the best. The developers in the team don't use the architecture much hence they have limited knowledge on the architecture. Meanwhile, the managers, who possess a competent understanding of the architecture, do not utilize it as frequently as the architects.

It was revealed through the interviews that the complexity of the architecture makes it challenging for users to locate the desired information in a timely manner. A common desire among all users for a more abstract and higher-level view of connections was expressed. The architects who regularly interact with the architecture expressed a preference for multiple abstract views of the same data, providing an overall illustration of the connections between components. The implementation of an API for the architecture was proposed as a solution by one of the developers.

During the interview, the stakeholders were asked about the tasks and questions they typically have when working with the architecture. The architects reported that they often begin by investigating the connections between two services, including all the interfaces that connect them. They may also need to understand the overall connections of the components, including its dependencies and the impact on other components.

The developers indicated that when making changes to the code for a service or interface, they must first understand the dependencies of the components. Similarly, when adding a new component, it is important to understand the overall structure of the architecture. Additionally, it is recognized that the specific needs of each stakeholder are dependent on the requirements of the situation.

It was determined that all stakeholders had a common expectation for the development of a tool that would be to provide dynamic search options, filtering capabilities, and visualization of the connections between all services within the architecture.

4.2 Experiment Results

The present study aims to evaluate the validity of a visualization tool for software architecture that has been developed based on the requirements of stakeholders. Three research questions are addressed in this evaluation: the time spent on the tool, the correctness of the users understanding of the software architecture and the subjective workload experienced by the users. The evaluation is conducted through a user study, and the collected data is organized and stored in an excel spreadsheet for analysis. The evaluation metric used in this study is 'User Performance'.

To outline, the scenarios discussed here are Scenario 1 is documentation with a UML component diagram, and Scenario 2 is the implementation of the visualization tool.

4.2.1 Time Result

This section will focus on the hypothesis associated with Research Question 1, which is the time spent by users in the different scenarios to answer the questions during the user study. The hypothesis will be tested using the Wilcoxon signed rank statistical method. The collected data is displayed in a grouped bar chart as shown in Figure 4.1. The grouped bar chart represents the data for both Scenario 1 and Scenario 2. Initially, the response time for each question was collected, however, for evaluation purposes, the total time that a user spent answering four questions in a given scenario was calculated by adding the time for each individual question.

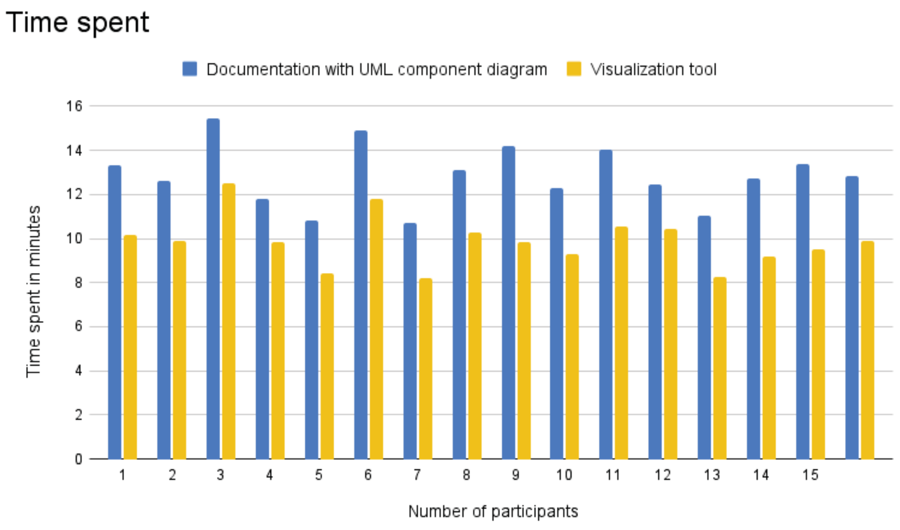


Figure 4.1: Grouped Bar Chart of Time Data

In the Grouped Bar Chart, the x-axis represents the number of participants, while the y-axis denotes the duration in minutes that each participant spent on each scenario. The box plot diagram in Figure 4.2 provides a visual representation of the distribution of the time data, allowing for a comprehensive understanding of the data's distribution.

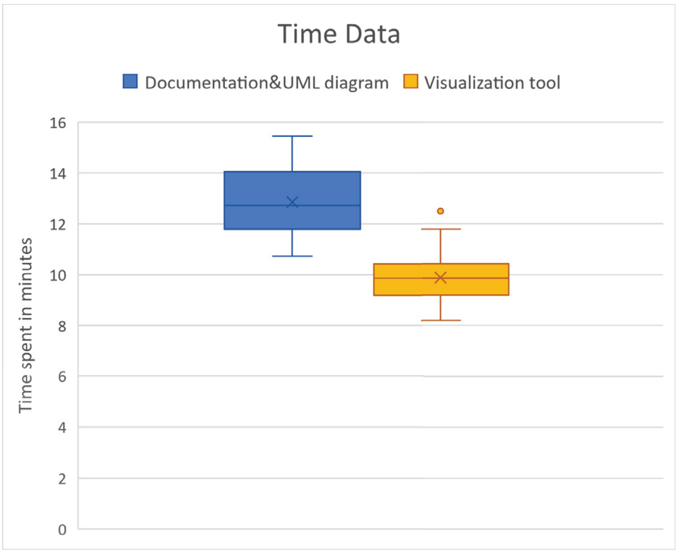
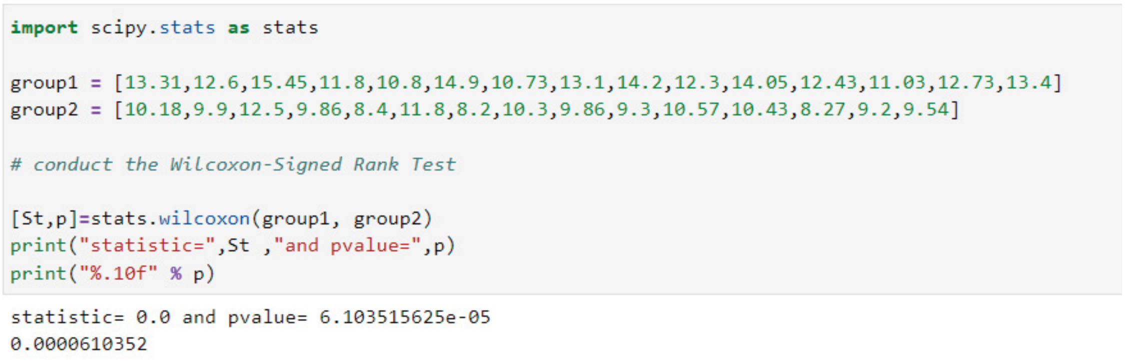


Figure 4.2: Box Plot Chart of Time Data

To test the hypothesis, a predefined function provided by python was utilized to calculate Wilcoxon singed ranked test, as illustrated in Figure 4.3.



```python
import scipy.stats as stats

group1 = [13.31,12.6,15.45,11.8,10.8,14.9,10.73,13.1,14.2,12.3,14.05,12.43,11.03,12.73,13.4]
group2 = [10.18,9.9,12.5,9.86,8.4,11.8,8.2,10.3,9.86,9.3,10.57,10.43,8.27,9.2,9.54]

# conduct the Wilcoxon-Signed Rank Test

[St,p]=stats.wilcoxon(group1, group2)
print("statistic=",St ,"and pvalue=",p)
print("%.10f" % p)
```

```
statistic= 0.0 and pvalue= 6.103515625e-05
0.0000610352
```

Figure 4.3: Wilcoxon Signed Rank Test for Time Data

To test the hypothesis, the p-value was used as a means of comparison. A way to test with the p-value is to say that the null hypothesis is rejected if the p-value is less than or equal to the alpha value. The value obtained from the python function is $6.10e^{-05}$ which is equivalent to 0.000061 which is less than 0.05(alpha value) hence it is evident that the null hypothesis is rejected and there is a significant difference in time spent by the users in the two scenarios.

4.2.2 Correctness Result

In this section, the focus will be on evaluating the hypothesis related to research question 2, which aims to determine whether the users have correctly answered architectural questions and have understood the architecture properly.

The procedure used in evaluating the correctness of the answers in the questionnaire involves assigning specific points to each question. The user is assigned these points if they provide a correct response. The first three questions have a similar level of complexity and are each assigned one point. The fourth question, which is comparatively more challenging with a longer answer, is assigned two points. If a user provides a complete and accurate response, they will receive two points. If their response is partially correct, they will receive one point. The scores of the users are obtained by aggregating the points earned for each question. The scoring system ranges from 0 to 5, with 5 being the highest score attainable.

The scores gained by all the users are shown in the group bar chart in Figure 4.4. The x-axis represents the number of participants, while the y-axis reflects the score obtained by each participant.

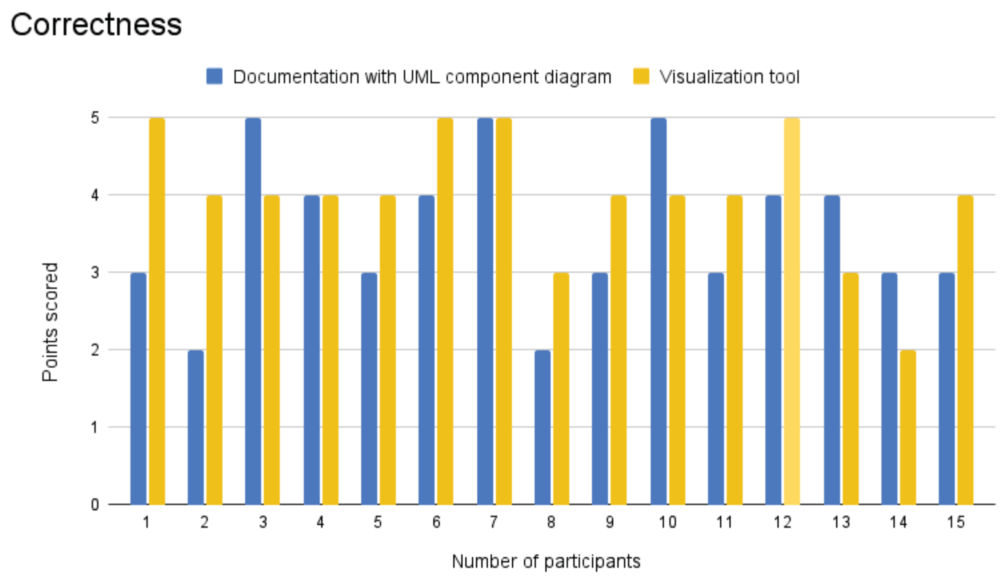


Figure 4.4: Grouped Bar Chart of Correctness Data

The box plot chart in Figure 4.5 serves to describe the distribution of the data, providing insight into its distribution pattern.

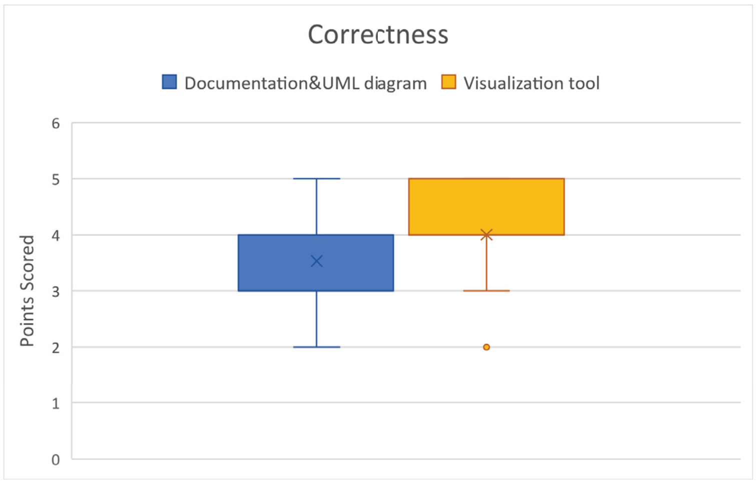
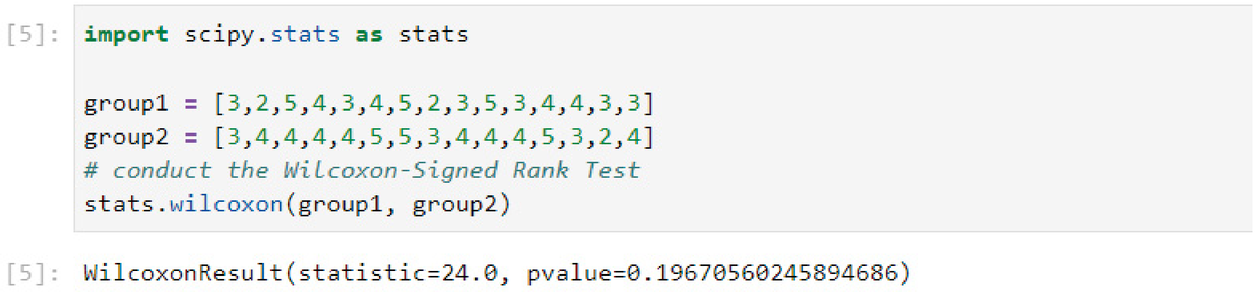


Figure 4.5: Box Plot Chart of Correctness Data

In order to validate the hypothesis, the Wilcoxon signed rank test values were obtained through the implementation of a pre-defined statistical library in Python, as depicted in Figure 4.6. The p-value obtained through the function is 0.1967. In this case, the obtained p-value is greater than the chosen level of significance (alpha) of 0.05. Therefore, it can be concluded that the null hypothesis is accepted.



```
[5]: import scipy.stats as stats

     group1 = [3,2,5,4,3,4,5,2,3,5,3,4,4,3,3]
     group2 = [3,4,4,4,4,5,5,3,4,4,4,5,3,2,4]
     # conduct the Wilcoxon-Signed Rank Test
     stats.wilcoxon(group1, group2)

[5]: WilcoxonResult(statistic=24.0, pvalue=0.19670560245894686)
```

Figure 4.6: Wilcoxon Signed Rank Test for Correctness Data

4.2.3 NASA-TLX Assessment Result

To test the hypothesis regarding research question 3, This NASA-TLX assessment test is performed. For this research, a 7-point scale was utilized in the NASA-TLX assessment form for the purpose of measuring the workload experienced by the users during the user study. users were asked to fill out the assessment form twice after completing a questionnaire for each scenario.

To analyze the results, each subjective question was evaluated individually based on the scores given by each user and to effectively present the results of the evaluation of the NASA TLX assessment tool, separate grouped bar charts and box-plot charts were generated for each of the six subjective questions. These charts serve as a visual representation of the data collected and to understand the data distribution through box-plot diagrams as represented in Figure 4.7, 4.8, 4.9, 4.10, 4.11, 4.12.

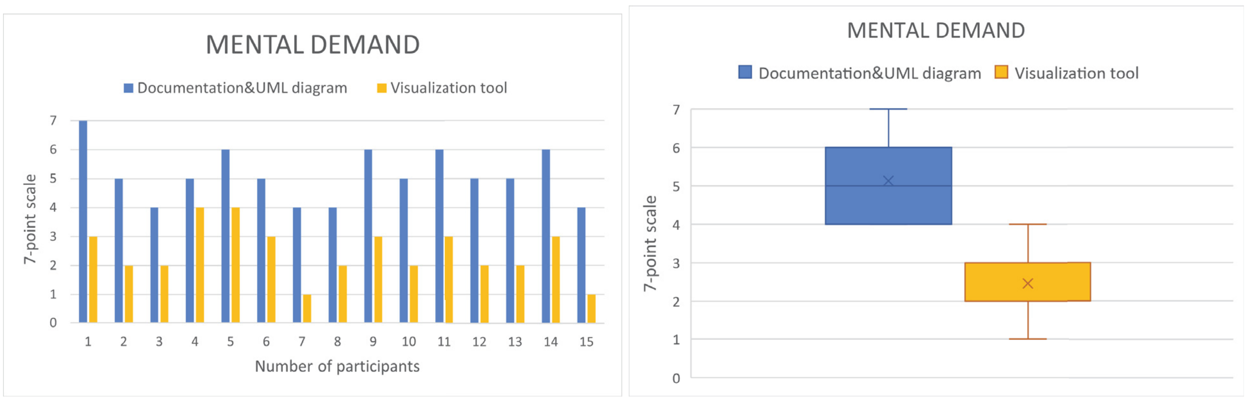


Figure 4.7: Grouped bar chart and Box Plot for Mental Demand

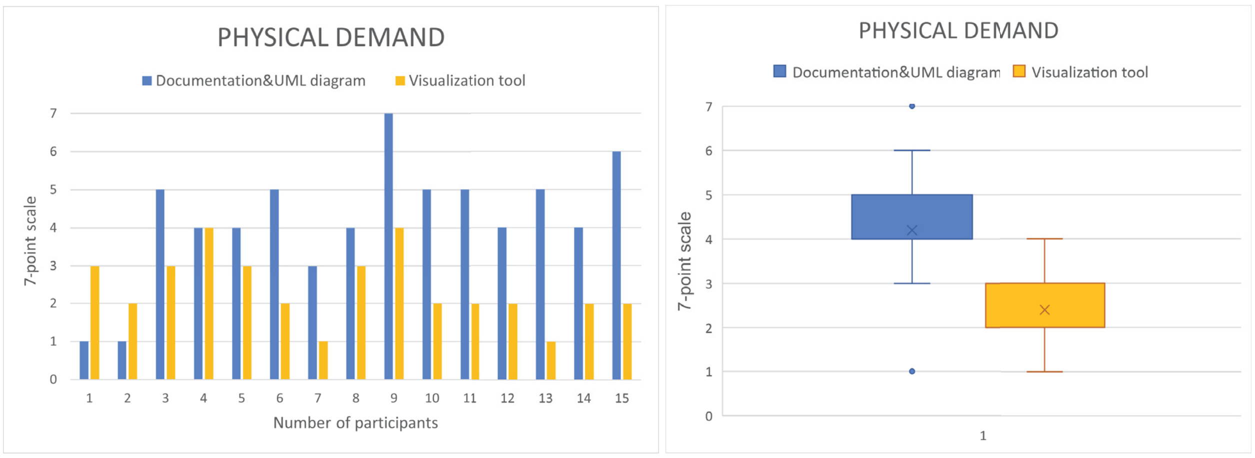


Figure 4.8: Grouped bar chart and Box Plot for Physical Demand

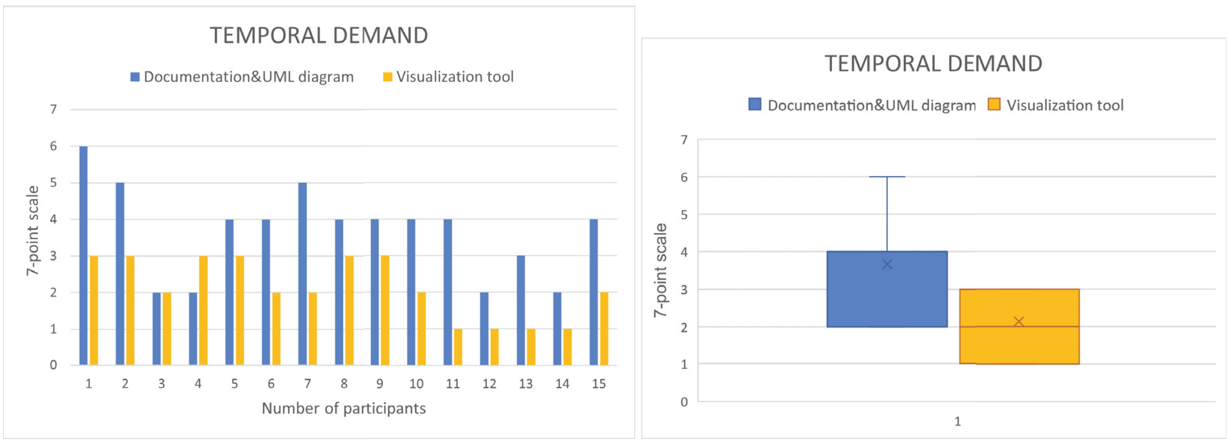


Figure 4.9: Grouped bar chart and Box Plot for Temporal Demand

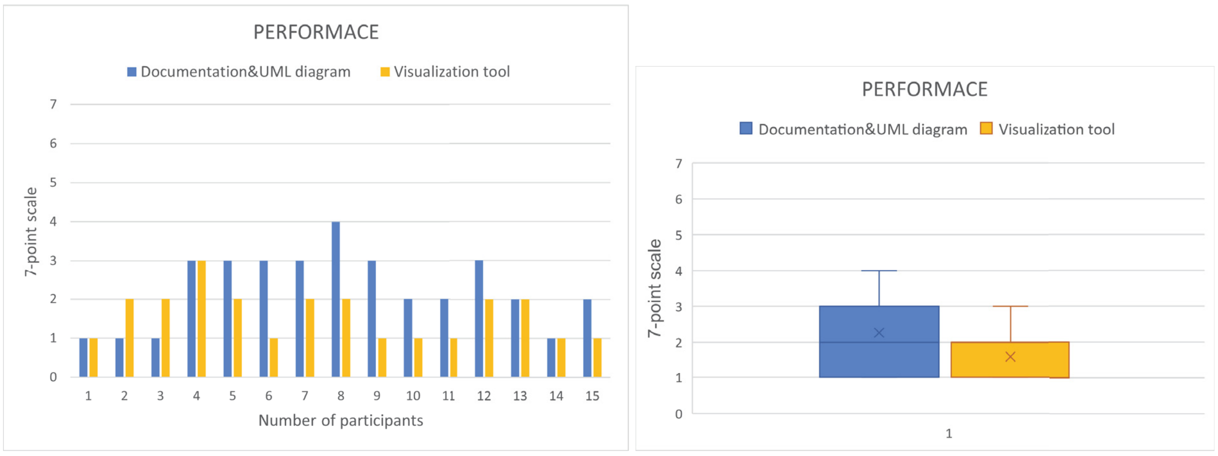


Figure 4.10: Grouped bar chart and Box Plot for Performance

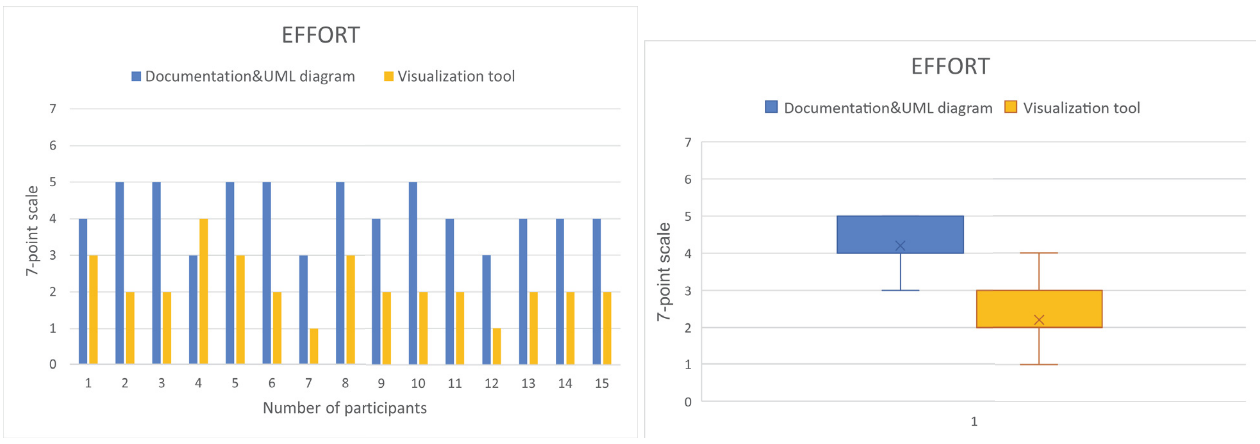


Figure 4.11: Grouped bar chart and Box Plot for Effort

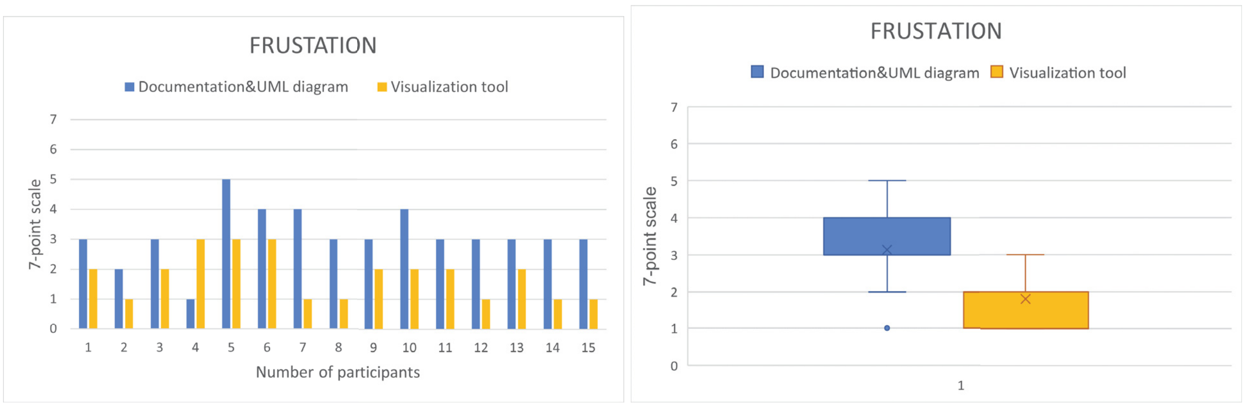
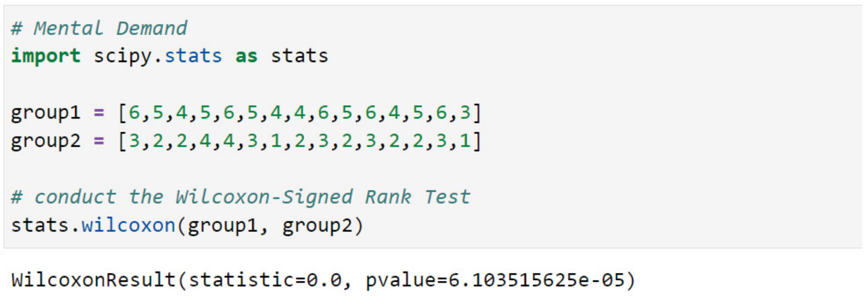


Figure 4.12: Grouped bar chart and Box Plot for Frustration

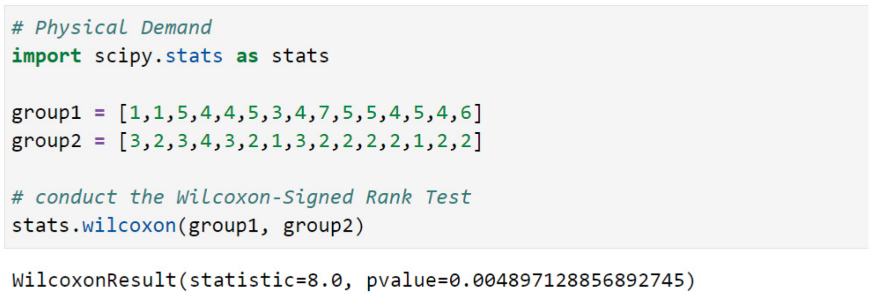
The data obtained from the six subjective questions were independently analyzed using the Wilcoxon signed-rank test. The results of this analysis for each of the six subjective questions are displayed in Figures 4.13,4.14,4.15,4.16,4.17,4.18.



```
# Mental Demand
import scipy.stats as stats

group1 = [6,5,4,5,6,5,4,4,6,5,6,4,5,6,3]
group2 = [3,2,2,4,4,3,1,2,3,2,3,2,2,3,1]

# conduct the Wilcoxon-Signed Rank Test
stats.wilcoxon(group1, group2)
```

```
WilcoxonResult(statistic=0.0, pvalue=6.103515625e-05)
```

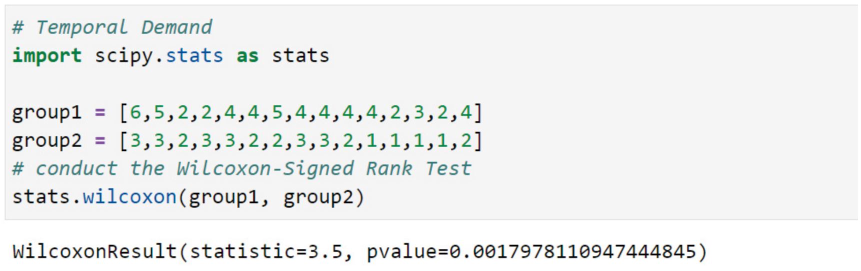
Figure 4.13: Wilcoxon Signed Rank Test for Mental Demand



```
# Physical Demand
import scipy.stats as stats

group1 = [1,1,5,4,4,5,3,4,7,5,5,4,5,4,6]
group2 = [3,2,3,4,3,2,1,3,2,2,2,2,1,2,2]

# conduct the Wilcoxon-Signed Rank Test
stats.wilcoxon(group1, group2)
```

```
WilcoxonResult(statistic=8.0, pvalue=0.004897128856892745)
```

Figure 4.14: Wilcoxon Signed Rank Test for Physical Demand



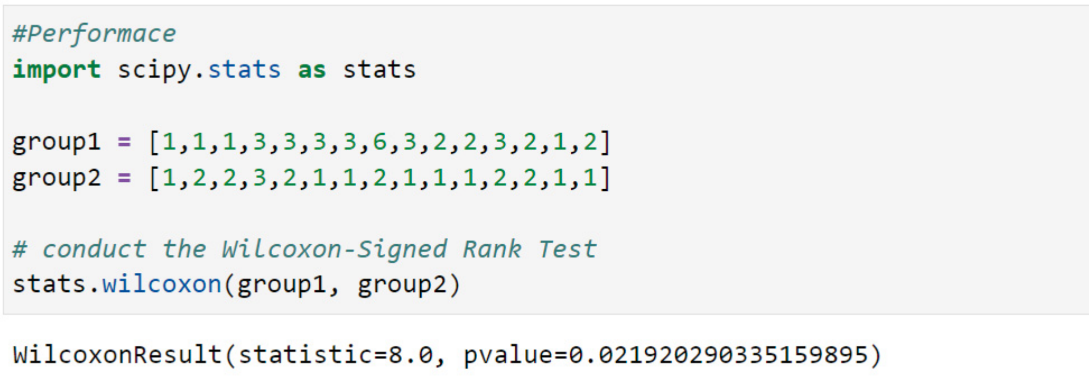
```
# Temporal Demand
import scipy.stats as stats

group1 = [6,5,2,2,4,4,5,4,4,4,4,2,3,2,4]
group2 = [3,3,2,3,3,2,2,3,3,2,1,1,1,1,2]
# conduct the Wilcoxon-Signed Rank Test
stats.wilcoxon(group1, group2)
```

```
WilcoxonResult(statistic=3.5, pvalue=0.0017978110947444845)
```

Figure 4.15: Wilcoxon Signed Rank Test for Temporal Demand

The results of the Wilcoxon signed-rank test for five subjective questions showed that the p-value was below the chosen alpha level of 0.05 and for performance questions, the p-value is more than the alpha value. As the subjective question on performance is not a crucial aspect of this research, It can be concluded that the null hypothesis for research question 3 is rejected. In other words, it can be concluded that there is a significant difference in the subjective ratings given by the user for each of the six questions.

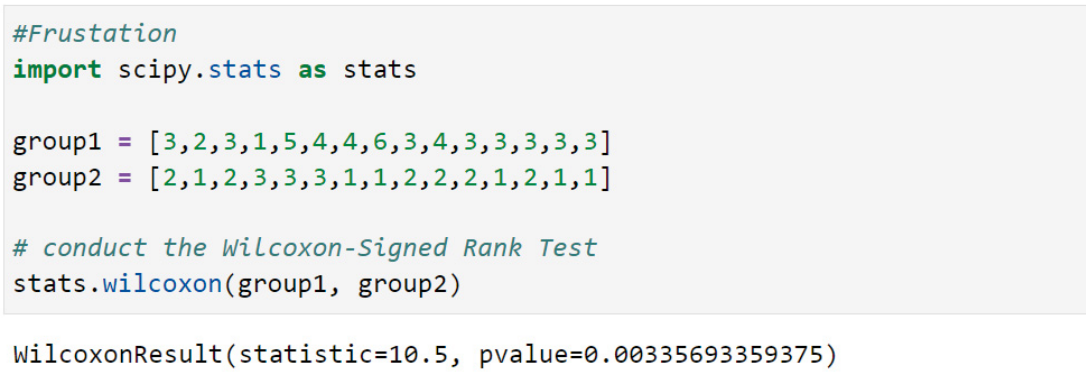
The bar charts in Figure 4.19 present the values chosen by various participants in the two scenarios on a seven-point scale in the NASA-TLX assessment form, and it clearly shows that the visualization tool had lower values in all the subjective questions. This demonstrates that the subjective workload experienced by users during the user study was lower with the implementation of the visualization tool compared to the documentation using a UML component diagram.



```python
#Performace
import scipy.stats as stats

group1 = [1,1,1,3,3,3,3,6,3,2,2,3,2,1,2]
group2 = [1,2,2,3,2,1,1,2,1,1,1,2,2,1,1]

# conduct the Wilcoxon-Signed Rank Test
stats.wilcoxon(group1, group2)
```

```
WilcoxonResult(statistic=8.0, pvalue=0.021920290335159895)
```

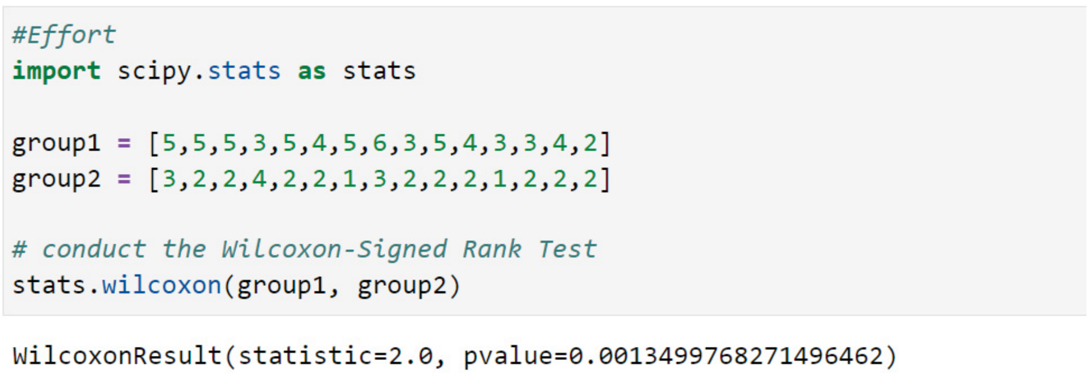
Figure 4.16: Wilcoxon Signed Rank Test for Performance



```python
#Frustation
import scipy.stats as stats

group1 = [3,2,3,1,5,4,4,6,3,4,3,3,3,3,3]
group2 = [2,1,2,3,3,3,1,1,2,2,2,1,2,1,1]

# conduct the Wilcoxon-Signed Rank Test
stats.wilcoxon(group1, group2)
```

```
WilcoxonResult(statistic=10.5, pvalue=0.00335693359375)
```

Figure 4.17: Wilcoxon Signed Rank Test for Frustration



```python
#Effort
import scipy.stats as stats

group1 = [5,5,5,3,5,4,5,6,3,5,4,3,3,4,2]
group2 = [3,2,2,4,2,2,1,3,2,2,2,1,2,2,2]

# conduct the Wilcoxon-Signed Rank Test
stats.wilcoxon(group1, group2)
```

```
WilcoxonResult(statistic=2.0, pvalue=0.0013499768271496462)
```

Figure 4.18: Wilcoxon Signed Rank Test for Effort

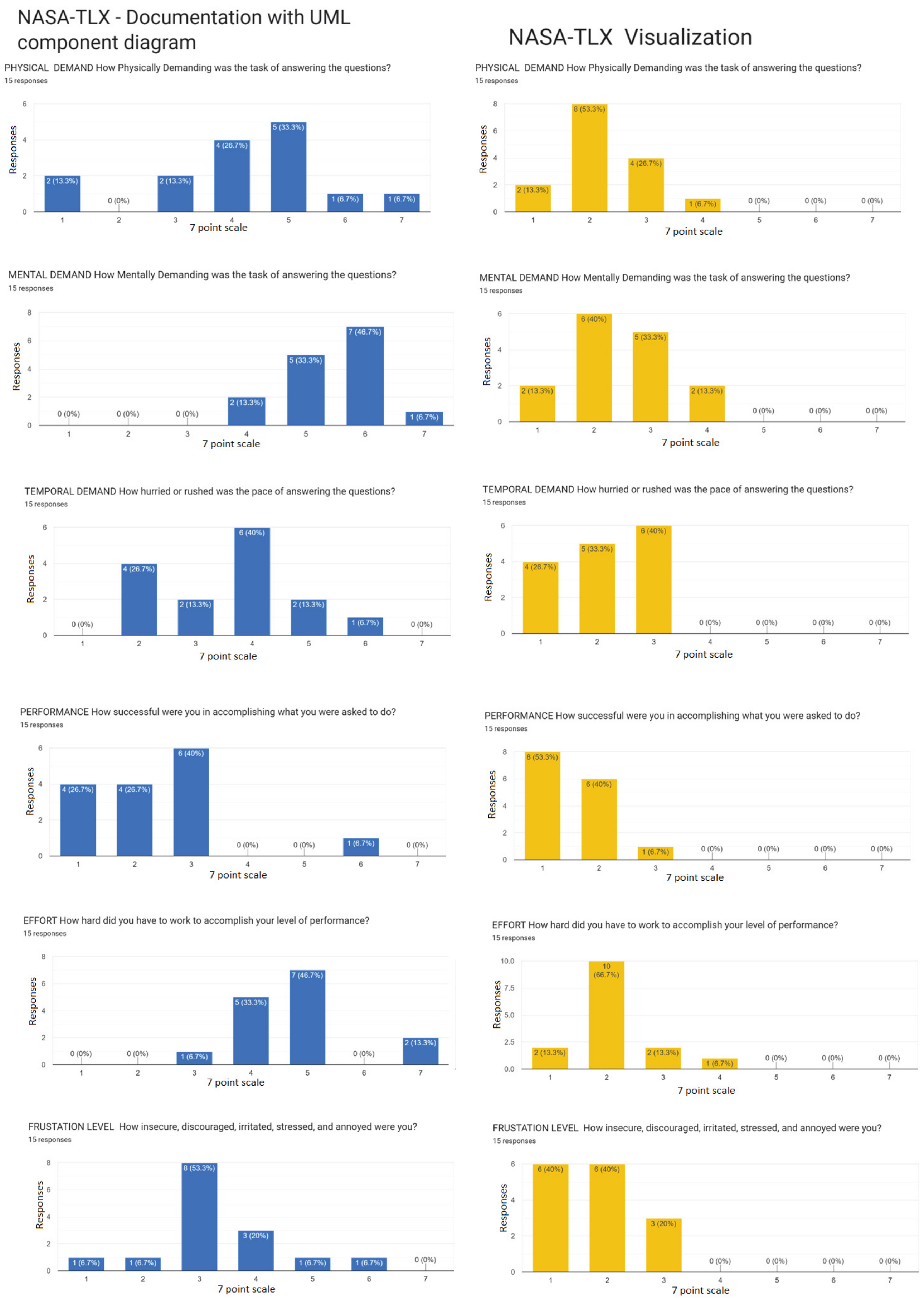


Figure 4.19: User Responses for NASA-TLX Assessment Test

Chapter 5
Discussion

This section includes discussions regarding the collected data, results, summarizing the thesis as a whole, and addressing threats to validity.

During the user study, the participants, who were employees of Ericsson, were given access to the software architecture documentation and the UML component diagram. However, all of the participants used only the UML component diagram in answering the questions due to their unfamiliarity with the architecture and difficulty in navigating and searching through the voluminous documentation, which they had not encountered previously.

At the start of the user study, some participants expressed some level of intimidation when answering questions using the documentation and UML component diagram. This was due to the lack of prior knowledge about the architecture and the complexity of the large number of components and complex UML component diagram.

When answering the first question, which involved searching for a service and listing its connected interfaces, the participants were able to quickly locate the services in the UML diagram. However, navigating to the connected interfaces proved challenging due to numerous overlapping lines, which made it difficult for the human eye to track and follow the line. This resulted in participants missing the connection and having to restart from the beginning. Later during the user study, it was observed that the last architectural in the questionnaire proved to be more challenging for the participants compared to other questions due to its increased level of complexity.

While answering the questions in the UML diagram, they wished that there was a search option because, after a while, they were a little exhausted trying to use the UML diagram to answer the question as they did not know the end goal of the thesis and one of the participant questioned the use of a UML diagram when the company already had a visualization because the participants did not know the end-user goal.

While using the implemented visualization tool, the participants made no complaints about how to use the tool, how to navigate, or anything else.The participants were mostly satisfied with the tool while using it. As this was the first time that all participants used the tool, it is anticipated that with regular usage, the tool will become increasingly convenient for all users of the system.

Among the group of participants who used the implemented visualization tool first, a decrease in stimulation was observed when they were later required to work with the UML diagram which required to carry out tasks manually. Conversely, a positive change in the experience was noted among those who initially worked with the UML diagram and later used the visualization tool.

Throughout this thesis, a consistent color scheme using blue and yellow has been employed in all figures, to ensure accessibility for individuals with color blindness. The blue color represents the documentation with the UML component diagram, while the yellow color represents the implemented visualization tool. This consistent use of colors makes it easier to understand the results and comparisons presented in the figures.

During the user study, some participants moved to the next question before thoroughly answering the current one. This could be due to their confidence that they have already answered all questions or because many of the participants are developers who may have limited understanding and infrequently use the architecture.

5.1 Research Question 1

For research question 1, results show a significant difference between the time spent by participants on the two scenarios, this statement is supported by showing statistical evidence using the Wilcoxon signed rank test. The time spent on the visualization tool by the users is reduced by 23 percent when compared to using the documentation with the UML component diagram.

In the collected data, it was observed that during the study, the participants took longer to answer the first question as they needed time to familiarize themselves with the tool and the questions. However, for subsequent questions, their response time was faster. In evaluating the results, the total time spent by the participants on answering all the questions in each scenario was considered.

To ensure a smooth and efficient user study, clear instructions and an introduction to the tool and the study as a whole were provided to the participants prior to the start of the study. Any questions or concerns raised by the participants during the study were addressed promptly to minimize potential delays in answering the questions.

5.2 Research Question 2

The results of research question 2 are that there is no statistical difference in the correctness while answering the questions in documentation with the UML component diagram and visualization tool. The participants were interested in doing the user study, they spent enough time to find the answers in both situations.

Throughout the user study, the primary objective of the participants was to answer the questions in the study accurately, regardless of the scenario, as they were not aware of the ultimate goal of the study and wanted to familiarize themselves with the tools provided. The results of the study showed that, in terms of the correctness of answers, there was not a significant difference between the two scenarios. Some participants performed well in both scenarios, scoring high points in both, while others performed moderately and a few performed below average in both. The poor performance of some participants may be due to a lack of understanding of the questions.

5.3 Research Question 3

To summarize the results of research question 3, the evaluation was carried out based on the six subjective questions and the values assigned by the participants during the user study. As discussed in section 4.2.3, the p-value obtained was less than the alpha value for five subjective questions and the p-value was greater than the alpha value in the performance question.

The highest difference in the subjective workload experience was in the mental demand category, with participants experiencing higher mental demand while using the UML component diagram and a comparably lower mental demand while using the implemented visualization tool. There was also a significant difference in physical demand, as participants felt they had to manually trace the lines in the diagram, resulting in some physical effort.

In the performance scale, the question asked was "How successful were you in accomplishing what you were asked to do?" and the majority of participants answered 'perfect' as they felt confident that they had answered the questions correctly.

Finally, for the remaining questions regarding temporal demand, effort, and frustration, participants rated their workload as being lower when using the implemented visualization tool.

5.4 Threats to Validity

In this thesis, there are some potential threats to validity to consider as this thesis is related to talking to multiple people, collecting data for the user study, and evaluating the data collected. This section is further divided into Internal, External and Construct Validity.

5.4.1 Internal Validity

It is understood that internal validity refers to the confidence that can be placed on the results that it accurately represents the real relationship between the variables in a study. The internal validity that might have an influence on this study are :

- Previous experience and educational background of the participants could have an impact on the results obtained in the study.
- In this study, the sample size was limited due to the challenges of gathering a larger number of participants within a confined timeframe within the organization.
- The user study was conducted as within the study, which means that the same participant had to answer the question in both scenarios, resulting in a longer user study. Because the participants are unaware of the thesis's end goal, they may become a little demotivated and exhausted at the end of the user study.
- While using the online questionnaire tool during the user study, the participants can see the time spent on each question in the top corner, for this reason,

participants may feel rushed even though it is clearly stated to them that they do not have limited time for each question and they do not need to be rushed.

- In this study, the participants were required to switch between answering questions and using the UML diagram or visualization tool, which may have resulted in increased time spent on each question.

5.4.2 External Validity

External validity defines as how adequately can the results be generalized to a large population, the potential external validity in this thesis is as follows:

- Because this is the first time the users have used the tool, and the documentation and UML diagram, the results may differ slightly from what the users will see when they use the system on a regular basis.
- Due to the limited sample size selected for this investigation, the possibility of external validity issues arises.

5.4.3 Construct Validity

The potential threat to the construct validity of this thesis are :

- One potential threat is that the UML component diagram used in this thesis is not provided by Ericsson but was created specifically for this thesis because to conduct a comparative study, the tools comparing should be on same level of comparability. The UML component diagram was developed in the most optimal way possible by having all the services in a row and all the interfaces in a row so that while searching for the services and interfaces, it does not get messed up and can be tracked properly. Since the architecture has many services and interfaces, it is challenging to place them in an optimal way.
- Another threat considered is that the architectural questions asked during the user study, may differ in real-life events according to the requirements. To mitigate this threat, the questions are formulated by talking to the architects on the team, understanding the use cases from the stakeholders from the preliminary interviews and formulating questions that are similar to the real ones.

Chapter 6

Conclusions and Future Work

An interactive visualization tool was developed and evaluated in this thesis with respective user performance metrics and also determined how much subjective workload is experienced by the users while using the documentation with UML diagram and visualization tool.

The implementation of graph-based visualization in this thesis has demonstrated its effectiveness and has proved to be a useful decision. This technique has enabled the visualization of the architecture at a high level of abstraction, providing a comprehensive overview of the architectural structure.

The basic requirements from Ericsson to have a tool with search and filtering options are achieved. In our implemented tool there is an efficient search box and also a highlighting option where components and their connections are highlighted when the user hovers over them. This enhances the usability of the tool and helps users easily locate and understand the components they are searching for.

The primary aim of this thesis was to create a tool that reduces the workload on the users of the system, which is accomplished in the implemented visualization tool, and The amount of subjective workload a user experiences is calculated by asking the user to fill out a NASA-TLX subjective assessment form during the user study.

The results of research question 1 clearly indicate that the time spent on the visualization tool is significantly reduced. the second research question revealed that users have a good understanding of the visualization tool and there is no significant difference in terms of correctness. The third research question showed that the subjective workload on users when using the visualization tool is significantly lower compared to using documentation and UML diagrams.

Based on these findings, it is clear that the implemented visualization tool outperforms the existing method at Ericsson, and it is also clear that the participants prefer the implemented visualization tool over the documentation and the UML component diagram.

In conclusion, the issue at Ericsson has been resolved through the development of a visualization tool and its subsequent evaluation to assure its validity.

6.1 Future Work

This thesis has the potential for further expansion. Currently, the data for the visualization tool is read from a JSON file, but when new components are added, this file must be manually updated. To automate this process, an API endpoint could be created so that the visualization tool can fetch updated data in real-time and remain constant without manual intervention.

Based on discussions with Ericsson architecture team members, it was discovered that the services are classified into subcategories internally. If the data was updated to include this information, there would be numerous new possibilities for visualization. For example, different levels of abstraction or hierarchies could be implemented, allowing users to only view the services and access further information as needed. This would help maintain clarity in the visualization as the architecture grows.

Given more time and an expanded scope of data, the visualization tool could be developed to integrate other architectures and be scaled for use by multiple teams across the company.

References

[1] M. Beck, J. Trümper, and J. Döllner, "A visual analysis and design tool for planning software reengineerings," *2011 6th International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT)*, pp. 1–8, 2011.

[2] M. T. Su, C. Hirsch, and J. Hosking, "Kaitorobase: Visual exploration of software architecture documents," in *2009 IEEE/ACM International Conference on Automated Software Engineering*, 2009, pp. 657–659.

[3] H. Byelas and A. Telea, "Visualizing metrics on areas of interest in software architecture diagrams," in *2009 IEEE Pacific Visualization Symposium*, 2009, pp. 33–40.

[4] A. Zalewski, S. Kijas, and D. Sokołowska, "Capturing architecture evolution with maps of architectural decisions 2.0," in *Software Architecture.* Springer Berlin Heidelberg, 2011, pp. 83–96. [Online]. Available: https://doi.org/10.1007%2F978-3-642-23798-0_9

[5] R. C. de Boer, P. Lago, A. Telea, and H. van Vliet, "Ontology-driven visualization of architectural design decisions," in *2009 Joint Working IEEE/IFIP Conference on Software Architecture European Conference on Software Architecture*, 2009, pp. 51–60.

[6] T. Panas, T. Epperly, D. Quinlan, A. Saebjornsen, and R. Vuduc, "Communicating software architecture using a unified single-view visualization," in *12th IEEE International Conference on Engineering Complex Computer Systems (ICECCS 2007)*, 2007, pp. 217–228.

[7] R. S. Pressman and B. R. Maxim, *Software engineering: a practitioner's approach*, 8th ed. New York: McGraw-Hill, 2015.

[8] I. Sommerville, *Software Engineering*, 9th ed. Harlow, England: Addison-Wesley, 2010.

[9] F. Bachmann, L. Bass, P. Clements, D. Garlan, J. Ivers, R. Little, P. Merson, R. Nord, and J. Stafford, *Documenting Software Architectures: Views and Beyond, Second Edition.* Addison-Wesley Professional, 2010.

[10] T. Munzner, *Visualization Analysis and Design.* Florida: A K Peters/CRC Press, 2015;2014;.

[11] C. Ware, *Information Visualization*, 2012.

[12] M. Shahin, P. Liang, and M. A. Babar, "A systematic review of software architecture visualization techniques," *Journal of Systems and Software*, vol. 94,

pp. 161–185, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121214000831

[13] S. Diehl, *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software*, 01 2007.

[14] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture, Volume 1, A System of Patterns.* Wiley, 1996.

[15] I. Nadareishvili, R. Mitra, M. McLarty, and M. Amundsen, *Microservice Architecture: Aligning Principles, Practices, and Culture.* Sebastopol: O'Reilly Media, Incorporated, 2016.

[16] K. Gallagher, A. Hatch, and M. Munro, "Software architecture visualization: An evaluation framework and its application," *Software Engineering, IEEE Transactions on*, vol. 34, pp. 260–270, 04 2008.

[17] Y. Ghanam and S. Carpendale, "A survey paper on software architecture visualization," 01 2008.

[18] M. Burch, M. Raschke, A. Zeyfang, and D. Weiskopf, "A scalable visualization for dynamic data in software system hierarchies," in *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, 2017, pp. 85–93.

[19] S. Bieliauskas and A. Schreiber, "A conversational user interface for software visualization," in *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, 2017, pp. 139–143.

[20] D. Seider, A. Schreiber, T. Marquardt, and M. Brüggemann, "Visualizing modules and dependencies of osgi-based applications," in *2016 IEEE Working Conference on Software Visualization (VISSOFT)*, 2016, pp. 96–100.

[21] B. Cornelissen, A. Zaidman, and A. van Deursen, "A controlled experiment for program comprehension through trace visualization," *IEEE Transactions on Software Engineering*, vol. 37, no. 3, pp. 341–355, 2011.

[22] B. Cornelissen, A. Zaidman, A. van Deursen, and B. van Rompaey, "Trace visualization for program comprehension: A controlled experiment," in *2009 IEEE 17th International Conference on Program Comprehension*, 2009, pp. 100–109.

[23] F. Fittkau, S. Finke, W. Hasselbring, and J. Waller, "Comparing trace visualizations for program comprehension through controlled experiments," in *2015 IEEE 23rd International Conference on Program Comprehension*, 2015, pp. 266–276.

[24] W. Hasselbring, A. Krause, and C. Zirkelbach, "Explorviz: Research on software visualization, comprehension and collaboration," *Software Impacts*, vol. 6, p. 100034, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2665963820300257

[25] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale, "Empirical studies in information visualization: Seven scenarios," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 9, pp. 1520–1536, 2012.

[26] F. Desimoni and L. Po, “Empirical evaluation of linked data visualization tools,” *Future Generation Computer Systems*, vol. 112, pp. 258–282, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X19303474

[27] A. G. Nuzzolese, V. Presutti, A. Gangemi, A. Musetti, and P. Ciancarini, “Aemoo: Exploring knowledge on the web,” in *Proceedings of the 5th Annual ACM Web Science Conference*, ser. WebSci ’13. New York, NY, USA: Association for Computing Machinery, 2013, p. 272–275. [Online]. Available: https://doi.org/10.1145/2464464.2464519

[28] I. Santana-Pérez, “Graphless: Using statistical analysis and heuristics for visualizing large datasets,” in *Proceedings of the Fourth International Workshop on Visualization and Interaction for Ontologies and Linked Data co-located with the 17th International Semantic Web Conference, VOILA@ISWC 2018, Monterey, CA, USA, October 8, 2018*, ser. CEUR Workshop Proceedings, V. Ivanova, P. Lambrix, S. Lohmann, and C. Pesquita, Eds., vol. 2187. CEUR-WS.org, 2018, pp. 1–12. [Online]. Available: http://ceur-ws.org/Vol-2187/paper1.pdf

[29] L. Po and D. Malvezzi, “High-level visualization over big linked data,” 10 2018.

[30] N. Bikakis, J. Liagouris, M. Krommyda, G. Papastefanatos, and T. Sellis, “graphVizdb: A scalable platform for interactive large graph visualization,” in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, may 2016. [Online]. Available: https://doi.org/10.1109%2Ficde.2016.7498340

[31] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, and A. Wessln, *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, 2012.

[32] P. Runeson and M. Höst, “Guidelines for conducting and reporting case study research in software engineering,” *Empirical Software Engineering*, vol. 14, pp. 131–164, April 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1519313.1519324

[33] T. Punter, M. Ciolkowski, B. Freimut, and I. John, “Conducting on-line surveys in software engineering,” in *2003 International Symposium on Empirical Software Engineering, 2003. ISESE 2003. Proceedings.*, 2003, pp. 80–88.

[34] F. Shull, J. Singer, and D. I. Sjøberg, *Guide to advanced empirical software engineering*. Springer, 2007.

[35] J. W. Creswell, *Research design: qualitative, quantitative, and mixed methods approaches*, fourth, international student ed. Los Angeles, Calif: SAGE, 2014.

[36] L. Banwell and G. Coulson, “Users and user study methodology: the jubilee project,” *Information research*, vol. 9, no. 2, pp. 9–2, 2004.

[37] P. Bourhis, J. L. Reutter, and D. Vrgoč, “Json: Data model and query languages,” *Information systems (Oxford)*, vol. 89, p. 101478, 2020.

[38] H. Rininsland, M. Heydt, and P. N. Castillo, *D3.js: cutting-edge data visualization : turn your raw data into real knowledge by creating and deploying complex data visualizations with D3.js*, 1st ed. PACKT Publishing, 2017.

[39] https://observablehq.com/@bstaats/graph-visualization-introduction.

[40] T. Munzner, *Visualization Analysis and Design*. Florida: A K Peters/CRC Press, 2015;2014;.

[41] A. Jedlitschka and D. Pfahl, "Reporting guidelines for controlled experiments in software engineering," in *2005 International Symposium on Empirical Software Engineering, 2005.*, 2005, pp. 10 pp.–.

[42] https://staruml.io/.

[43] G. Charness, U. Gneezy, and M. A. Kuhn, "Experimental methods: Between-subject and within-subject design," *Journal of Economic Behavior Organization*, vol. 81, no. 1, pp. 1–8, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167268111002289

[44]

[45]

[46] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wilcoxon.html.

[47] https://humansystems.arc.nasa.gov/groups/tlx/index.php.

[48] L. Colligan, H. W. Potts, C. T. Finn, and R. A. Sinkin, "Cognitive workload changes for nurses transitioning from a legacy system with paper documentation to a commercial electronic health record," *International Journal of Medical Informatics*, vol. 84, no. 7, pp. 469–476, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1386505615000635

[49] J. K. Nuamah, Y. Seong, S. Jiang, E. Park, and D. Mountjoy, "Evaluating effectiveness of information visualizations using cognitive fit theory: A neuroergonomics approach," *Applied Ergonomics*, vol. 88, p. 103173, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0003687020301277