Visualisation of data from IoT systems
- A case study of a prototyping tool for data visualisations

Visualisering av data från sakernas internet system

Jonathan Anderson

Supervisor:
Jonas Wallgren (Linköping university)
Marcus Svensson (Attentec)

Examiner:
Ola Leifler
Upphovsrätt

Copyright
The publishers will keep this document online on the Internet – or its possible replacement – for a period of 25 years starting from the date of publication barring exceptional circumstances. The online availability of the document implies permanent permission for anyone to read, to download, or to print out single copies for his/hers own use and to use it unchanged for non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional upon the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility. According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement. For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its www home page: http://www.ep.liu.se/.

© Jonathan Anderson
Abstract

The client in this study, Attentec, has seen an increase in the demand for services connected to Internet of things systems. This study is therefore examining if there is a tool that can be used to build fast prototype visualisations of data from IoT systems to use as a tool in their daily work.

The study started with an initial phase with two parts. The first part was to get better knowledge of Attentec and derive requirements for the tool and the second part was a comparison of prototyping tools for aiding in development of data visualisations. Apache Zeppelin was chosen as the most versatile and suitable tool matching the criteria defined together with Attentec. Following the initial phase a pre-study containing interviews to collect empirical data on how visualisations and IoT projects had been implemented previously at Attentec were performed. This lead to the conclusion that geospatial data and NoSQL databases were common for IoT projects. A technical investigation was conducted on Apache Zeppelin to answer if there were any limits in using the tool for characteristics common in IoT system. This investigation lead to the conclusion that there was no support for plotting data on a map.

The first implementation phase implemented support for geospatial data by adding a visualisation plug-in that plotted data on a map. The implementation phase was followed by an evaluation phase in which 5 participants performed tasks with Apache Zeppelin to evaluate the perceived usability of the tool. The evaluation was performed using a System Usability Scale and a Summed Usability Metric as well as interviews with the participants to find where improvements could be made.

From the evaluation three main problems were discovered, the import and mapping of data, more feature on the map visualisation plug-in and the creation of database queries. The first two were chosen for the second iteration where a script for generating the code to import data was developed as well as improvements to the geospatial visualisation plug-in. A second evaluation was performed after the changes were made using similar tasks as in the first to see if the usability was improved between the two evaluations. The results of the Summed Usability Metric improved on all tasks and the System Usability Scale showed no significant change. In the interviews with the participants they all responded that the perceived usability had improved between the two evaluations suggesting some improvement.
Acknowledgements

A special thanks to Johanna for your ideas, for answering all my questions and for all the proofreading. A big thanks Ola for answering all my emails during and after office hours. I would also like to thank my opponent Ludwig Wikblad for the constructive comments during the process. Last but not least I would like do direct a big thanks to everyone at Attentec for the support during the thesis and for taking part in interviews and evaluations.

Jonathan Anderson
Linköping, June 2017
# Contents

## Abstract

## Acknowledgments

## Contents

## List of Figures

## List of Tables

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Aim</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Research question</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Delimitations</td>
<td>3</td>
</tr>
<tr>
<td>1.4.1 Data Limitations</td>
<td>3</td>
</tr>
<tr>
<td>1.4.2 Development Process</td>
<td>3</td>
</tr>
<tr>
<td>1.4.3 Security</td>
<td>3</td>
</tr>
<tr>
<td><strong>2 Theoretical framework</strong></td>
<td>4</td>
</tr>
<tr>
<td>2.1 Internet of Things</td>
<td>4</td>
</tr>
<tr>
<td>2.1.1 Sensors and actuator</td>
<td>5</td>
</tr>
<tr>
<td>2.1.2 Processing unit</td>
<td>5</td>
</tr>
<tr>
<td>2.1.3 Communication</td>
<td>6</td>
</tr>
<tr>
<td>2.1.4 Storage</td>
<td>6</td>
</tr>
<tr>
<td>2.1.5 Visualisation and Analysis</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Data</td>
<td>6</td>
</tr>
<tr>
<td>2.3 Data from IoT</td>
<td>8</td>
</tr>
<tr>
<td>2.4 Big Data</td>
<td>8</td>
</tr>
<tr>
<td>2.5 Visualisation techniques</td>
<td>9</td>
</tr>
<tr>
<td>2.5.1 Pixel-Oriented visualisation</td>
<td>9</td>
</tr>
<tr>
<td>2.5.2 Geometric Projection</td>
<td>10</td>
</tr>
<tr>
<td>2.5.3 Icon-based visualisation</td>
<td>10</td>
</tr>
<tr>
<td>2.5.4 Hierarchical visualisation</td>
<td>11</td>
</tr>
<tr>
<td>2.5.5 Graph-based or complex data visualisation</td>
<td>11</td>
</tr>
<tr>
<td>2.6 Usability</td>
<td>11</td>
</tr>
<tr>
<td>2.6.1 Usability guidelines</td>
<td>12</td>
</tr>
<tr>
<td>2.6.2 Usability evaluation</td>
<td>14</td>
</tr>
<tr>
<td>2.7 Requirement elicitation</td>
<td>15</td>
</tr>
<tr>
<td><strong>3 Related work</strong></td>
<td>17</td>
</tr>
<tr>
<td><strong>4 Method</strong></td>
<td>19</td>
</tr>
</tbody>
</table>
4.1 Study structure ......................................................... 19
4.2 Method and methodologies ........................................ 20
   4.2.1 Scientific Approach ........................................ 20
   4.2.2 Scientific Reasoning ........................................ 20
   4.2.3 Research Methodologies .................................... 20
   4.2.4 Questionnaires .............................................. 21
   4.2.5 Interviews .................................................. 21
   4.2.6 Literature .................................................. 21
   4.2.7 Triangulation ................................................ 21
   4.2.8 Controlled experiments .................................... 22
4.3 Method Model ..................................................... 22
   4.3.1 Initial phase ................................................ 23
   4.3.2 Theoretical Study ........................................... 24
   4.3.3 Pre-study ................................................... 24
   4.3.4 Implementation, iteration 1 ................................ 26
   4.3.5 Evaluation, iteration 1 ...................................... 26
   4.3.6 Implementation, iteration 2 ................................ 28
   4.3.7 Evaluation, iteration 2 ...................................... 28
4.4 Method criticism .................................................. 28
5 Results ............................................................... 30
   5.1 Initial phase .................................................... 30
      5.1.1 Background and planning ................................ 30
      5.1.2 Visualisation tools ........................................ 31
      5.1.3 The chosen tool .......................................... 34
   5.2 Pre-study ....................................................... 35
      5.2.1 Interviews ................................................. 35
      5.2.2 Technical evaluation ..................................... 38
   5.3 Implementation iteration 1 ................................... 39
   5.4 Evaluation iteration 1 ......................................... 39
      5.4.1 Task creation .............................................. 40
      5.4.2 SUS evaluation ........................................... 41
      5.4.3 SUM evaluation .......................................... 42
      5.4.4 Interviews ................................................. 43
   5.5 Implementation iteration 2 ................................... 47
      5.5.1 Import from MongoDB .................................... 47
      5.5.2 Improving the geospatial visualisation plug-in ........... 47
      5.5.3 Improving code editor .................................... 49
   5.6 Evaluation iteration 2 ......................................... 49
      5.6.1 Task creation .............................................. 50
      5.6.2 SUS evaluation ............................................ 52
      5.6.3 SUM evaluation .......................................... 52
      5.6.4 Interviews ................................................. 53
   5.7 Compilation of the evaluations ............................... 57
      5.7.1 SUS ....................................................... 57
      5.7.2 SUM ....................................................... 58
      5.7.3 Interviews ................................................. 59
6 Discussion .......................................................... 60
   6.1 Prototyping ..................................................... 60
   6.2 Usability ........................................................ 61
   6.3 Choosing the tool .............................................. 61
   6.4 Evaluations ..................................................... 61
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Main components of an IoT system</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>Example of a pixel oriented visualisation</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Examples of a geometric projection visualisations</td>
<td>10</td>
</tr>
<tr>
<td>2.4</td>
<td>Example of an icon based visualisation with Chernoff faces</td>
<td>11</td>
</tr>
<tr>
<td>2.5</td>
<td>Example of a treemap</td>
<td>11</td>
</tr>
<tr>
<td>2.6</td>
<td>Example of a tag cloud</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>Method model</td>
<td>22</td>
</tr>
<tr>
<td>5.1</td>
<td>Image of the plug-in developed during implementation 1</td>
<td>39</td>
</tr>
<tr>
<td>5.2</td>
<td>Data from iteration 1</td>
<td>40</td>
</tr>
<tr>
<td>5.3</td>
<td>Image of the code generation plug-in developed during implementation 2</td>
<td>48</td>
</tr>
<tr>
<td>5.4</td>
<td>Image of the code generation plug-in developed during implementation 2</td>
<td>49</td>
</tr>
<tr>
<td>5.5</td>
<td>Image of the code generation plug-in developed during implementation 2</td>
<td>49</td>
</tr>
<tr>
<td>5.6</td>
<td>Example data iteration 2</td>
<td>51</td>
</tr>
</tbody>
</table>
List of Tables

4.1 Interviews during pre-study. ............................................................... 25
5.1 Search hits. ......................................................................................... 33
5.2 SUS evaluation answers. ................................................................. 42
5.3 Goal time for each task in iteration 1. ......................................... 43
5.4 Evaluation results. ................................................................. 44
5.5 Calculated SUM. ................................................................. 45
5.6 SUS evaluation answers. ................................................................. 52
5.7 Goal time for each task in iteration 2. ......................................... 53
5.8 Evaluation results table. ................................................................. 54
5.9 Calculated SUM. ................................................................. 55
5.10 SUS evaluation answers iteration 1 and 2. ................................... 57
5.11 Calculated SUM iteration 1 and 2. ......................................................... 58
1 Introduction

In the following chapter the background and aim for this study will be presented to illustrate why this topic is interesting to investigate. The research questions that this study will try to answer will then be presented followed by the delimitations chosen.

1.1 Motivation

The client, Attentec, has seen an increase in the demand for services connected to Internet of things systems, IoT systems. The task is therefore to examine if there is a tool that can be used to build fast prototype visualisations of data from IoT systems so that Attentec can use it as a tool in their daily work.

Internet of things, IoT for short, is a concept which means that the objects surrounding us are increasingly connected to the internet. These connected devices can use sensors, actuators or both sensors and actuators, together with a network connection to communicate with the internet. This can for example be sensors in street lamps that reports if the lamp is working and uploads that data to the internet. By visualising the data from the street lamps the maintenance team will get information on when a lamp needs to be replaced.

A more formal definition from the Internet of Things Global Standards Initiative, IoT-GSI, is “a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.” [1]. In the definition IoT-GSI describe IoT as an infrastructure connecting things to enable advanced systems using our current and new technologies. When talking about IoT, it can sometimes be confusing to know what things are since it is such a broad term. Things in IoT are anything that can be connected to a processing unit and to the Internet [2].

Many of the IoT devices generate large amounts of data that are sent and collected over the internet. A single data entry in this raw collected data might not be understandable or meaningful by itself. This is what makes it important to visualise the collected data. The amount of data might also make it hard to see trends by just looking at the data. Another consequence with the large amount of collected data is that the visualisation needs to be able to scale to work with a large data set. By processing and visualising the collected data it can be made meaningful. A meaningful visualisation should present the data so that it becomes
understandable and so that the receiver is able to interpret the data and draw conclusions or make decisions from it.

The collected data can have different characteristics. It can be geodata, it can be a single value for example representing the temperature, it can be a video stream or it can be a combination of multiple types for example. The data can also be formatted in different ways while transmitted for example in JSON format, XML or CSV.

The risk with failing to visualise the data in a timely manner is to miss business opportunities and with it comes the risk of losing money and opportunities. A good visualisation on the other hand, enables the user of the system to conduct analyses that for example can improve a product or streamline operations.

Visualisations can be used as a tool to solve numerous problems where large amounts of data are collected and needs to be analysed. For example, it can be used to analyse meter data from an IoT environment to assist decision makers in predicting electricity consumption or in the area of retail and logistics to improve the shipping experience for the customer [3].

In the development of software it is important to early on include and present something to the customer to get feedback. This is so important that it is the first principle in the Agile Manifesto: “Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.” [4]. This early customer involvement is also one of the most valuable links between the customer and the developers in custom projects as found by Keil and Carmel [5]. By presenting a visualisation prototype to the customer that on the spot can be modified to present different angles of the data can save time in development and satisfy the customer. The development team will also get fast feedback from the customer on what they want to change with the visualisation.

In this study a comparison of tools for aiding in the development of prototypes for customers requiring data-intensive visualisation applications will be made. The tool most suitable to Attentecs needs will be chosen from the comparison for a case study in which it will be examined on the perceived usability for software developers. It will also be investigated if there are any characteristics in data from IoT systems that can make it more or less appropriate to use the chosen tool as a prototyping tool for visualisation.

1.2 Aim

This study aims is to select and evaluate a tool for aiding in the development of prototypes for visualisation of IoT data. The first evaluation will examine the strengths and weaknesses of the tool regarding what characteristics in data that makes it suitable to use. This tool will then be improved according to Attentecs needs in two iterations to see if it can be made more useful from a developer’s perspective. The evaluation for these two iterations will be how the perceived usability of the tool is from a developer’s perspective.

1.3 Research question

- What are the limits and benefits with using a prototyping tool for aiding in the development of visualisations of data from Internet of things systems regarding data characteristics and the developers perceived usability of the system?

The research question can be divided into sub questions whose answers in turn will answer the main research question.

1. Which tool is most suitable for using as a prototyping tool for data visualisations regarding Attentecs needs?

2. Are there any limits in using the chosen tool for the case study for some characteristics or formats of data?
3. Is the chosen tool for the case study usable from a developer’s perspective on the tools perceived usability?

4. Is it possible to make any improvements to the chosen tool to improve the perceived usability from a developer’s point of view?

1.4 Delimitations

The following subsections will present the delimitations made in this study.

1.4.1 Data Limitations

This study will not examine how to collect the data or how to access other potential data from sensors. It will also limit the scope to using the data that is already collected and stored from the IoT system. The data will due to secrecy in customer data come from free open data sources. The data sources used in the study will be chosen to be representative of data from IoT systems.

1.4.2 Development Process

From a development point of view this study will not go in to depth on what development approaches or methods exist and can be used for the study. An iterative process will be used to enable improvements and evaluations during the development in the study.

1.4.3 Security

Since the data is transferred over the internet it is of importance to keep the data secure, and security in IoT could be a subject for another study. The scope of this study will however not go into depth in the security issues in IoT systems.
2 Theoretical framework

In this chapter a theoretical framework will be presented with theories regarding the topic of this thesis to be used as a scientific base in this study. Theories on the subject of IoT will be presented first continuing on to the subject of data and data from internet of things. Big data and data visualisations conclude the more technical sections. Lastly a section about usability and a section about requirement elicitation will be presented.

2.1 Internet of Things

The idea of controlling things remotely has been around since the early 1990s and the birth of the internet [6]. However, at this time the technical conditions needed for IoT were not in place. In 1999 the term Internet of things was made popular by Kevin Ashton, executive director of the Auto-ID Centre at MIT [6]. The Internet of Things Global Standards Initiative, IoT-GSI, defines internet of things as “a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.” [1]. Today around 6.4 billion things are connected to the internet according to the research institute Gartner and by the year 2020 they estimate that the number will be higher than 20 billion devices [7]. One of the key features of the connected things is that they need to be manufactured at a low cost so that the increased value of the product gained from being connected to the internet outweighs the cost.

The application of IoT can be divided in to different categories. Gubbi et al. categorises them in the following four: Enterprise, Utilities, Personal and home and Mobile [8]. Another categorisation from Suresh et al. is: Smart environment, Domestic applications, Industrial Applications, Security and Emergencies and Logistics and transport [6]. Even though they use different names, both Gubbi et al. and Suresh et al. have quite similar categories when they are compared to each other. Personal and home and Domestic applications are referring to IoT in a home or consumer setting like smart homes, smart watches etc. Enterprise and Industrial applications are referring to IoT in a work or industrial environment where an example could be the tracking of products across a production line. Mobile and Logistics and transports is also mostly the same domain that covers IoT in the transport sector. An example here could be improved vehicle routing due to the usage of real-time traffic data. Utilities
2.1. Internet of Things

and Smart environment are both referring to applications in the infrastructure and the smart city which includes smart electrical grids and improved monitoring of water networks. Suresh et al. includes Security and Emergencies as an additional category which includes surveillance and detection of emergencies. One thing that both categorisations have in common is that the data from one domain could be useful in another domain. The different domains might however have different needs about how to use the data since their goals differ. The city might be interested in data about the amount of traffic to plan for maintenance and new roads while the same data could be useful for route planning in a logistics company. In this study the data used for the evaluation comes from the category containing Utilities and Smart environment applications.

The main components of an IoT system can be seen in Figure 2.1. When creating an IoT device the first component is the sensor or actuator that receives data from the surroundings or that, from data, affects the surrounding. The sensor or actuator is connected to the second part which is the processing unit. The processing unit is responsible through a trans-receiver for the third part that is the communication of the data. How to communicate the data is an important decision that depends on the type of thing and the type of data to be transmitted. The next part is the collection and storage of the data and the last step is an analysis or visualisation of the data.

In the following sections the different parts of the IoT system will be explained briefly one by one. Since this study is focusing on how to perform a visualisation of the data and the understandability of the visualisation these parts will be described in more detail in later sections.

2.1.1 Sensors and actuator

A sensor detects information or change in its environment and outputs that as data. A few examples of sensors are thermometers, GPS, accelerometers and so on. Whereas a sensor detects information and then sends it on, an actuator receives data and then affects its surroundings according to that data. Some examples of actuators are electric motors, electromagnets and hydraulic cylinders.

2.1.2 Processing unit

When it comes to the processing unit there are mainly two different approaches, either to use one of the common platforms or to use a microcontroller. The use of common platforms, like the Arduino platform or the Raspberry Pi platform brings a couple of benefits to the area. Communities around the platforms develop and maintain the libraries and software for the platforms and therefore a developer does not need to have knowledge of the underlying
Data

The increased use of common platforms has lowered the entry level of developing IoT applications since the developers no longer need to have a highly specialised knowledge to be able to develop this kind of applications. This in turn is contributing to the rapid development of IoT systems. Microcontrollers or system on a chip as they also might be called are often cheaper to use on a larger scale and they are typically smaller and consume less power. They are however often more restricted in the programmable language and better suited for single purpose applications.

Communication

One of the first things to be classified as IoT devices was Radio-Frequency IDentification (RFID) tags which use radio frequency to communicate. These RFID tags can be either passive or active but active tags requires a power source. Passive sensors can only be read at close distances while the active have a longer range. Near Field Communications (NFC) and Wireless Sensor and Actuator Networks (WSAN) are two other ways of communicating.

The IEEE 802.15.4 standard is used by many commercial sensor network solutions for example Zigbee, 6LoWPan, Thread, WiFi etc. The standard defines the physical and MAC layers for communications in wireless personal area networks (WPAN). Other common techniques for communication include Bluetooth Low-Energy (BLE) which has become popular among wearable products and Cellular GSM/3G/4G/5G that can be used when communication over a long distance is needed.

Storage

The collected data must be stored somewhere for it to be usable. This storage includes both how the data is stored and where it is stored. For the data to be accessible some kind of database is usually used to store the data after it has been collected.

NoSQL, Not Only SQL, is a common type of databases when working with big data from IoT. SQL, Structured Query Language, is used in relational databases which makes NoSQL databases not only relational. An example of a NoSQL system is MongoDB.

When it comes to where to store the collected data the cloud has the benefit of it being easily accessible but local storage is also a possibility. One important factor when working with storage of data from IoT systems is that the storage solution chosen must be able to handle the large amounts of data generated from most IoT systems.

Visualisation and Analysis

The visualisation of data is an important part of an IoT application that is needed to extract meaningful information from the raw data. "Data analytics" and "GIS based visualisation" are mentioned by both Gubbi et al. and Bhuvaneswari and Porkodi as challenges in the area of IoT among a few other challenges. Mohsen et al. talks about that data visualisations can be difficult due to the large amounts of data and due to data being unstructured or semi structured. Since data visualisations are a vital part of this study, they are further investigated in section 2.5.

Data

Data is what links the sensors in the physical world to the computers and internet where the analysis is made. The transformation from raw data to a visualisation can be split into four phases: raw data, data tables, visual structures and views. The last phase, views, handles how to interact with the visualisation and will therefore be covered in section 2.5 below. Raw data is the received unmodified data from the data source. One way to classify the raw
data is by its value. The values can be Nominal (unordered set), Ordinal (ordered set) and Quantitative (numeric range or value) [19]. The raw data is then transformed into data tables. This often involves a change in the information where for example the data could be sorted, converted, min and max values, geodata and timestamps could be added to the data. [18] This step is often what adds the different characteristics to the data, and what defines the format in which the data will be stored and accessed.

In the next step the data tables are mapped to visual structures. A mapping is said to be better than another mapping if it is less time-consuming to interpret. [18] A classification of data in visual structures was made by Bertin [20], He classifies data into two areas, spatial and retinal variables. Spatial variables are, as the name suggests, coordinates in the visualisation. Retinal variables on the other hand contain information about a visual element. Retinal variables can include size, texture, colour etc. [20]

Ziemkiewicz and Kosara agrees that Bertin have contributed greatly to the research field of visualisation design and practices but also identifies some issues with his classification. One of these issues is that the usage of certain colours and shapes can make the receiver read in something to the graph that was not intended. An example of such usage is that when visualising an organisation: If the organisation are visualised using circles the organisation can be thought to be organic and cooperative by the interpreter while squares symbolises isolation and rigidity. [21]

One way of categorising different visualisations is by looking on how the spatial and retinal variables change.

To be able to categorise the visualisation the kind of reference system used becomes important. A visualisations reference system contains the elements’ relationship relevant to the task. A stable reference system could for example be the geographical positions of cities while an example of an unstable reference could be the position of cars on a road since they keep changing. The spatial variables can be divided in to four categories: Fixed, Mutable, Create and Create & Delete depending on how they behave. [22]

It is possible to chose the category of spatial variables most suited for the visualisation. The choice of category is based on if the comparisons should be made based on timestamps in the data or data at a given state and if the reference system is stable or not. [22]

**Fixed**, where the elements are kept in the same place and the number of elements are fixed. This is preferred when creating a time comparison with a stable reference system. An example could be a visualisation of sensors measuring the road temperature on fixed positions on a map.

**Mutable**, where the elements might change position but the number of elements are fixed. This is preferred when creating a time comparison where the reference system is not stable. This could for example be a visualisation of the position of taxi cabs within a company.

**Create**, same as mutable but new elements can be added. This is preferred when creating comparisons within the current state using a stable reference system. For example a visualisation of data traffic to a server where every new request is added to the visualisation.

**Create & Delete**, same as Create but the elements can also be removed. This is preferred when creating comparisons within the current state but the reference system is not stable. An example could be a visualisation of connected phone calls at a telephone exchange where new calls are added to the visualisation and removed when they end. [22]

The retinal variables can also be divided in to four categories in a similar way where the categories are: Immutable, Known Scale, Extreme bins and Mutable Scale. Based on the knowledge about the temporal variation in data and the scale it is possible to chose the category of retinal variables best suited for the visualisation. [22]


2.3 Data from IoT

Data from IoT systems can be very different and have different requirements depending on from what kind of system it is. One area where there can be differences in the requirements for different things is the transfer of the data. For weather data it might not matter if the data takes a few seconds to reach the receiver but for real-time traffic data it might be crucial that the data is transferred immediately. Another difference in these examples is the form of the data that is transmitted.

Even though there can be differences in the data depending on the kind IoT system there are a few characteristics in the data that are common in most IoT systems: Karkouch et al. lists seven characteristics as common in data from IoT systems.

**Uncertain and noisy data**, that can come from multiple factors: cheap sensors (failing, bad values, low accuracy), loss of connection, etc.

**Voluminous and distributed data**, due to large number of sensors in multiple locations.

**Smooth variation**, which means that the value makes small changes between the sensor readings.

**Continuous or sampled data**, data that can take any value in a continuous range. Sampled data is continuous data that have been measured at specified time intervals.

**Correlations**, when data points have a dependence on other data points.

**Periodicity**, when data points follow a reoccurring pattern, for example the day/night cycle.

**Markovian behaviour**, that is when the value at a given point is a function of the previous value.

2.4 Big Data

The amount of data generated has increased over the past two decades and it is estimated to double at least every two years in the near future. The fast growing segment of IoT is one of the reasons as to why the amount of data generated continues to grow since this type of system generates large amounts of data. This increased amount of generated data creates several problems when it comes to storing, using and managing the collected data sets. Since the data sets from IoT systems often are immense it is closely connected to big data. Big data is a term that is used to describe large amounts of data that often is unstructured which fits well with data from IoT systems.

There exist many different definitions of what big data is and what it is that differentiates it from other kinds of data. The Apache Hadoop project was one of the first to define it
2.5. Visualisation techniques

and they defined it as “datasets which could not be captured, managed, and processed by
general computers within an acceptable scope.” [23]

Doug Laney described challenges that follows with increased data amounts in 2001 which
can be seen as one of the first definitions of big data. He used a model, also known as the
3Vs model, which described the increase in turns of increased volume, velocity and variety
of data. [24]

In 2011 McKinsey & Company released a report in which they define big data as the
following, “Big data refers to datasets whose size is beyond the ability of typical database
software tools to capture, store, manage and analyse.” [25].

A month later International Data Corporation, IDC, defined big data as the following,
“Big data technologies describe a new generation of technologies and architectures, designed
to economically extract value from very large volumes of a wide variety of data, by enabling
high-velocity capture, discovery, and/or analysis.” [26]. This definition from IDC vaguely
resembles Doug Laney’s definition with the addition of an extra V for value.

What all the mentioned definitions have in common is that big data is big data sets that
requires specialised techniques to handle and make useful. These big data sets have in com-
mon that they consist of a large volume of data but it also has increased velocity and variety.
The increase velocity is the required speed of data processing and variety refers to that there
are numerous types of data. All this together also gives big data a higher value than other
types of data.

It is important to evaluate the data quality of big data before performing any analytics
on it as to not draw the wrong conclusions based on errors in the data [27]. A few com-
mon reasons for low data quality are: ad-hoc instrumentation which could be changes in
algorithms that makes the comparison invalid, inconsistent or missing data and unclear
ownership of data and analysis [28].

2.5 Visualisation techniques

The large amount of data collected from IoT systems might be meaningless if the data points
are analysed one by one. Instead, when used in large quantities the data points could high-
light important characteristics in the data. Therefore, for the collected data to be valuable
it has to be presented in a context and in a way so that the receiver can interpret it. Most
visualisations are also dynamic since new data for the visualisation is continuously collected
which requires even more consideration on how to visualise it.

There are five different categories in which techniques for visualisation can be divided:
Pixel-Oriented, Geometric Projection, Icon-based, Hierarchical and Graph-based or Visualis-
ing of complex data and relations [29, 30]. Each of these categories in turn contains a number
of different ways to practically implement a visualisation [30]. In the following sections these
techniques will be described one by one with some practical applications as example.

Since data from IoT systems seldom contains only a single value, most of the data records
used in the examples below will have several attributes. For data records with several at-
tributes each of the attributes stored corresponds to a dimension in the sections below. A
data record may for example contain information such as a timestamp, a value and a user
which in that case corresponds to three dimensions.

2.5.1 Pixel-Oriented visualisation

In a pixel-oriented visualisation each data point gets a pixel in a fixed position and the value
decides the colour. Because of this, it is only possible to visualise data in one dimension
[29]. To use a pixel-oriented visualisation for multidimensional data subwindows can be used
and displayed side by side where the related data points are located in the same position
2.5. Visualisation techniques

in all graphs [30]. Relations, dependencies or correlations could be detected between the dimensions by displaying them side by side [30]. There are a few different design choices that must be made: how should the pixels be arranged in the window, the mapping between data and colours and the shape and order of the subwindows [30]. An example of a pixel-oriented visualisation can be seen in figure 2.2 where the pixels in the left chart could visualise the age of a population and the right chart their satisfaction with their occupation on a numerical scale where each number is represented with a shade of grey.

![Figure 2.2: Example of a pixel oriented visualisation.](image)

2.5.2 Geometric Projection

Geometric projections are based on the same principles as pixel-oriented visualisations but without the fixed position in more than one dimension [29]. Instead of displaying a data point in a predefined position it is instead placed in a two-dimensional plane or three-dimensional room. This makes it possible to show up to four dimensions in a geometric projection by using a colour for the pixel as the fourth dimension [29]. The common charts belong to this category, among them bar charts, line charts, scatter plots and pie charts as seen in 2.3. Other examples of geometric projections are scatter plot matrices, landscapes and parallel coordinates [30].

![Figure 2.3: Examples of a geometric projection visualisations.](image)

2.5.3 Icon-based visualisation

In this visualisation technique the data points are abstracted by using icons to represent the data [29]. Icon-based visualisation could for example use Chernoff faces that uses slightly different facial features to express different variables that together creates facial expressions. Humans are good at differentiating between these expressions thus making this a way of displaying multidimensional data in one image. [31] In figure 2.4 an example of an icon based visualisation with Chernoff faces can be seen.
2.5.4 Hierarchical visualisation

In a Hierarchical visualisation the dimensions are partitioned and each subset is then visualised hierarchically [29]. An example of a Hierarchical visualisation could be a country that has multiple regions that have multiple cities which then would translate in to a hierarchical view. Different ways to visualise hierarchical data are through treemaps or sunburst graphs [32, 33]. In a treemap the drawable area is divided in to boxes of different sizes and colours corresponding to the data values [32]. An example of a treemap can be seen in figure 2.5, in the figure each area could for example represent the population of a country. Sunburst graphs are best described as layered pie charts where the different levels of the layers indicates the hierarchical depth. [33]

2.5.5 Graph-based or complex data visualisation

Visualising complex data and relations can be done with tag clouds where the data, often in the form of keywords, are displayed with different colours and sizes depending on the usage area and usage rate [29]. An example of a tag cloud can be seen in figure 2.6. For relational data a circular network diagram could be used where the nodes are placed around the circle and lines of different sizes and lengths are drawn to display relations in the data. [29]

2.6 Usability

There are numerous definitions of usability. Usability can be seen as a quality attribute in interactive products [34]. The interactive product is in this study referring to the visualisation tool evaluated. According to Ottersten and Berndtsson the usability is high if it fulfils the expectations from users with similar purposes in using the system [34]. Usability can also be defined as the ease of use of any technological interface that allows interaction between humans and machines [35]. The ISO-definition of usability (ISO 9541:11) as printed in “Användbarhet i praktiken” by Ottersten and Berndtsson follows: “Usability is the extent to which a product can be used by
2.6. Usability

Figure 2.6: Example of a tag cloud.

specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.\[34\].

From the ISO definition it is clear that usability depends on the users of the particular system and their impression of the three attributes effectiveness, efficiency and satisfaction in the specified context of the system. Effectiveness can be likened to in which extent the system does the right task and efficiency can be likened to if it does the tasks in an efficient way.

The perceived usability is not only dependent on the system. Instead, usability is highly depending on the person and the behaviour and personal differences among the users as well as the cultural diversity, gender, ages etc. will affect what usability entail for different users \[36\]. Brooke went as far as to say that usability in itself does not have a meaning, instead it has to be put into a context to be meaningful \[37\].

The main benefits from a system with good usability are increased productivity \[36, 34, 38\], a shorter learning period \[36, 34, 38\], fewer errors by the users \[36\], retention of the system over time \[36, 38\] and user satisfaction \[36, 34, 38\].

There are often some trade-offs between the different characteristics such as a higher productivity at the cost of a longer learning period. Depending on the users needs, different characteristics have a higher or lower priority in designing the system. \[36\]

Usability is present in many quality models, one of the earliest is Factors In Software Quality where it is an important factor in the quality of a software \[39\]. Other quality models containing usability are Boehm, FURPS, Dromey, and ISO 317/2001 \[40\].

McCall, Richards and Walters \[39\] divide usability into three criteria, training, communicativeness and operability. They state that a higher usability could result in a trade off on efficiency. Usability could also affect other quality factors positively for example correctness, reliability, integrity, maintainability, testability and flexibility. \[39\]

The aspects of usability defined by ANSI 2001 and ISO 9241 are effectiveness, efficiency and satisfaction. Effectiveness is most commonly used to be measured by completion rates and errors while efficiency is measured by time on task and satisfaction is usually measured with a questionnaire \[41\].

2.6.1 Usability guidelines

In this section guidelines from the literature on how to design a system with usability in mind will be presented. The focus will be on an overhead perspective and not go in to too many details. Ottersten and Berndtsson cover in “Användbarhet i praktiken” \[34\] the process of developing a good and usable software. Shneiderman and Plaisant \[36\] then continues to present design tips for the development phase. Both sections are then backed-up by other articles on the subject to get a more reliable knowledge base.
Ottersten and Berndtsson state in “Användbarhet i praktiken” [34] that the first step towards usability is to understand the customers’ intention with the product and map the impact by looking at the expected usefulness, customer segment and usage goals. When knowledge has been gained about what to develop and to whom there is a need to continue with an analysis of the targeted customer segment to be able to focus the development on the most important parts that the end user wants. In the customer segment it is important to learn who the users are, what the users need and try to gather knowledge about the users. [34]

The information gathered in the previous two activities can then be used to create a requirement specification. The requirement specification should contain target groups, benefits or improvements from the product, use cases, goals and requirements. From the requirement specification it is then time to start developing the application. Before starting to write code, start with a design document that specifies the components of the system and how to navigate them. A functional design that explains what actions that can take place and what information that is provided is also good to have. Lastly choose the visual theme of colours, sizes, fonts, audio etc. [34]

Shneiderman and Plaisant presents what they call “The Eight Golden Rules of interface design” [36]. These eight rules are presented below and are useful tips and guidelines in the design process. Sajedi et al. [42] also describes some guidelines that loosely corresponds to Shneiderman and Plaisant’s Eight Golden Rules and is used as a complement in the description below and that adds some practical tips in some cases. There have been numerous earlier similar works on the same subject. One is the earlier work by Nielsen 1993 in Usability engineering where he describes his 10 Heuristics [43].

**Strive for consistency**, this is important for the user to be able recognise the system and how to use it. To get consistency the sequence of interactions needed to perform tasks should be standardised so that the user gets familiar with how to perform the tasks. One way of doing this is to keep to the standards, for example use radio buttons for single choice and checkboxes for multiple and not the other way round. As much as possible of the content should also be fixed to a theme of colours, sizes, fonts, audio, wording, placement, etc. [36] A system with high consistency lets the user predict how the system works and lowers the training needed to use the system [42].

**Cater to universal usability**, which is important so that both advanced and novice users can use the system. Both advanced and novice users should be able to use the system easily and efficiently. When the user gets more familiar with the system, the number of interactions shall be reduced by having shortcuts for tasks to increase the pace of interaction. [36, 42]

**Offer informative feedback**, this means that all actions should be followed by feedback from the system [36]. If an action takes more than 2 seconds the user shall be informed using a progress bar or similar that indicates that the system is working [42].

**Design dialogs to yield closure**, this is important so the user knows where in the process they are. Action sequences should be organised in to a start, middle and end phase. An example could be a web store where at the start the user puts the item in the cart, the middle is the display of the cart and filling out all info and the end is the “thanks for the order” message. By having these sequences it gives the user the satisfaction of accomplishment. [36]

**Prevent errors**, which is quite self-explanatory since the users are more likely to find the system usable if they do not conduct any errors. The system shall be designed to prevent the user from making serious errors and if the user makes an error the system shall have a simple, from the users’ perspective, mechanism for handling the error [36]. One way to prevent errors is to use tool tips and first usage guides [42].
2.6. Usability

Permit easy reversal of actions, since the user should never feel lost in the system. Being able to reverse an action performed in the system encourages exploration of the system which leads to faster learning and better performance of different tasks [36, 42].

Support internal locus of control, so that the user easily can navigate the system. The user shall feel in control over the system and the system shall behave as expected. [36]

Reduce short-term memory load, so that the user does not need to go back to double check things. The interface shall be designed so that the user does not have to remember information from one screen and use on another. [36]

The guidelines from this chapter will be used and referred to in the method chapter and explained in a more practical setting referring to the evaluation in this study.

2.6.2 Usability evaluation

Since there are no absolute measurement of usability the measurements has to be dependent on the context in the same way as the usability is [37]. Therefore, the most common types of usability evaluations are questionnaires, user testing, heuristic evaluations, interviews, and thinking aloud protocols according to a systematic mapping review published in 2016 [44].

A commonly used questionnaire is the SUS evaluation described later in this section. A heuristic evaluation means that the system is evaluated based on predefined principles [45]. Some examples of usability heuristics that can be used are the Eight golden rules by Shneiderman and Plaisant or Nielsen’s 10 Heuristics. Four possible measurements for evaluating usability is: time to learn, retention over time, a user’s error rate and user satisfaction [42].

System Usability Scale

A common method of evaluating the usability of a system is to use the System Usability Scale, commonly abbreviated to SUS. This is a method that allows a low cost assessment of the system usability. This scale consists of 10 questions concerning the system where the respondents can rank their experience of the system on a scale from 1 to 5. On this scale 1 means that the user strongly disagree with the statement while a 5 means that the user strongly agree with the statement. The questions in SUS are the following:

1. I think that I would like to use this system frequently
2. I found the system unnecessarily complex
3. I thought the system was easy to use
4. I think that I would need the support of a technical person to be able to use this system
5. I found the various functions in this system were well integrated
6. I thought there was too much inconsistency in this system
7. I would imagine that most people would learn to use this system very quickly
8. I found the system very cumbersome to use
9. I felt very confident using the system
10. I needed to learn a lot of things before I could get going with this system

These questions cover all aspects of usability since it is a very wide concept. The respondent should give their first thought as an answer and not think about the question for a long time for the questionnaire to be effective. [37]
The respondent gives a value of 1 to 5 on all questions before the scoring of the SUS can be calculated. This is done so that for question 1, 3, 5, 7 and 9 the score contribution is the respondent’s answer minus one. For question 2, 4, 6, 8, and 10 on the other hand the score contribution is 5 minus the scale position. The total score for all the questions are then added together and multiplied with 2.5 to give the systems SUS score. This score ranges from 0 to 100. \[37\] This single digit answer is quite useful when comparing competitive alternatives or iterative versions to determine which has the highest usability.

A proposed scale presented in “An Empirical Evaluation of the System Usability Scale” \[46\] states that a score bellow 50 is generally not acceptable, a score between 50 and 70 indicates some problems and a score above 70 is generally an acceptable software. \[46\]

### Summed Usability Metric

Another method that tries to measure the whole definition of usability as defined by ANSI and ISO is the summed usability metric, SUM. This method uses four metrics, task completion, error counts, task times and satisfaction scores to evaluate the usability of all three dimensions together. The respondent is asked to conduct a series of tasks while the test leader is keeping track of the time, completion rate and errors. The user’s satisfaction is measured with a questionnaire where the user can rank their experience on a scale from 1 to 5. \[41\] The questions about task satisfaction are asked after completion of each task and are questions about the difficulty of the task, rating of the amount of time to complete this task and task satisfaction. \[47\] The following questions are suggested by Sauro and Kindlund in Using a single usability metric (SUM) to compare the usability of competing products: \[47\]

1. How would you describe how difficult or easy it was to complete this task? From 1, Very Difficult, to 5, Very Easy.
2. How satisfied are you with using this application to complete this task? From 1, Very Unsatisfied, to 5, Very Satisfied.
3. How would you rate the amount of time it took to complete this task? From 1, Too Much Time, to 5, Very Little Time.

These questions are asked after each task and a mean value for all the tasks are then calculated to show the user’s satisfaction with the system. All four metrics are then standardised one by one. For the continuous data, time and satisfaction, this is done through subtracting the mean value from a predefined specification limit and then divided by the standard deviation.

Nielsen and Levy found that the mean value of system satisfaction for their supposedly usable system scored 4.1 as a mean value and 4.0 as a median value for a scale between 1-5 in “Measuring usability: preference vs. performance” \[48\]. Therefore, Sauro and Kindlund \[49\] propose 4 as the specification limit for satisfaction.

For the two discrete values task completion and errors the standardisation is done by dividing the failures with the number of possible failures, the number of attempts. The standardised mean value of the four metrics, task completion, error counts, task times and satisfaction are then calculated to give the usability of the whole system. This standardised and summed usability metric can be used to analyse the usability context as well as for comparison between different systems. To enable a more in depth analysis the metric can be expanded to cover other metrics concerning usability, for example click counts for task completion. \[41\]

### 2.7 Requirement elicitation

Eliciting requirements is an important step in the software development process. It is important for both the developers, to know what to implement, and for the users to know what to
expect. The process of defining the requirements should lead to the customer or user having a better understanding of their needs and constraints and to be able to evaluate solution alternatives. For the developers the process should produce a specification for the problem that should be solved so that both the customer and the developer have the same vision of the solution to the problem. \[50\]

Carrizo \[51\] identified six constructs that are important for a good requirement elicitation technique. These constructs are listed below.

- Efficiency
- Adequacy
- Quantity of information
- Effectiveness
- Quality of information
- Others

When comparing researchers and practitioners the practitioner holds all six constructs equally important while researchers rates efficiency lower and quality of information higher than the practitioners. \[51\]

Common problems when eliciting requirements are poor communication, resistance to new ideas, obstacle to communication on technical matters and problems with different perspectives between the developers and customers. \[50\]

One common technique to elicit requirements is prototyping. Prototyping is a way of reducing risk by having an early focus on what is feasible and to help identify real requirements. Prototypes are disposable and it is generally okay to not have any focus on otherwise important characteristics such as reliability, robustness and efficiency. \[50\] Andriole \[52\], advocates that throwaway prototyping is cost-effective and improves the requirement specification. He also states that prototyping should be responsive to the changing expectations and needs of the customer. \[52\]

There are two types of prototypes, low-fidelity prototypes that generally have limited functionality and are used to depict concepts and high-fidelity prototypes that have complete functionality and are interactive. A low-fidelity prototype is often created quickly and used to communicate and educate and not to train or test while a high-fidelity prototype is useful for exploration and testing. Rudd et al. \[53\] argues that both high- and low-fidelity prototypes are useful in the design process. \[53\]

There are some advantages and disadvantages with both high- and low-fidelity prototypes. Low-fidelity prototypes often have a lower development cost, can evaluate multiple design concepts and Proof-of-concept \[53, 54\]. The disadvantages with low-fidelity prototypes are that they have limited error checking \[53, 54\], poor detailed specification to code to \[53, 54\] and a limited utility after the requirements are established \[53\].

High-fidelity prototypes have the advantages of complete functionality \[53, 54\], being fully interactive \[53, 54\] serves as a living specification \[53, 54\] and as a marketing and sales tool \[53\]. The disadvantages of high-fidelity prototypes are that they are more time-consuming and expensive to develop \[53, 54\], they might blind the users to major representational flaws \[54\] and management may think they are real \[54\].
In this section relevant parts of two systematic reviews on studies in information visualisation will be presented followed by two case studies that have performed similar studies as this study. The focus will be on how to evaluate the usability of information visualisation tools. The intent with this section is to describe how similar studies have been performed as a motivation for the evaluation methods presented in the method chapter.

In “Empirical Studies in Information Visualization: Seven Scenarios” by Lam et al. [55] a literature review was conducted on studies in information visualisation. These studies were categorised in seven different scenarios depending on the goal of the study. The scenario that matches this study the closest is the user experience, UE, scenario.

In that scenario the evaluations of the visualisations rely on people’s subjective feedback and opinions on the tool. Commonly studies in this scenario examine how people react to a visualisation regardless of if the visualisation is an initial sketch, a working prototype, or a finished product. The goal of these studies is to get knowledge on how the visualisation supports the intended use case from the participant’s point of view and to find requirements and needs.

Lam et al. lists five questions that are common to address in a UE study which are all touched upon to some degree in the study of this thesis. All of the questions are listed below:

1. What features are seen as useful?
2. What features are missing?
3. How can features be reworked to improve the supported work processes?
4. Are there limitations of the current system which would hinder its adoption?
5. Is the tool understandable and can it be learned?

Common evaluation methods in UE are informal evaluations, usability tests and field observations. An informal evaluation is as the name suggest an informal evaluation where the user does not have a set task but simply uses with the system to test it. A usability test is when the participants perform a set of predefined tasks followed by questionnaires or interviews to evaluate the system. Field observations are similar to usability test but the study is
performed in a real-world setting, to examine how the user uses the system. 34 percent of the studies examined in the study by Lam et al. were of the type UE, which also was the largest group. Between one and five participants were most common in UE evaluations. [55]

Another study about visualisation evaluations is “A Systematic Review on the Practice of Evaluating Visualization” by Isenberg et al. [56] which is based on the study by Lam et al. presented above. In this review the authors enhance the scenarios developed by Isenberg et al. by adding a new scenario. This scenarios is called qualitative result inspection, QRI, and is based on UE but without the user. By excluding the user this scenario lets the reader interpret the results themselves. The new classification QRI is used in 46 percent of all scenarios and lowers UE to 9.3 percent off all scenarios. Since the study in this thesis is based on evaluations with users it can still be classified as a UE scenario. The study by Isenberg et al. mentions a common pitfall in UE evaluations namely that:

“While subjective positive feedback from experts on a new technique or tool is encouraging and can be a valid tool of evaluation, simply stating ‘... and they really liked it...’ is not sufficient.”.

One way to mitigate such answers is to also ask questions that criticises the tool. Other parts of a study that is important for the credibility and validity mentioned by Isenberg et al. to be lacking in many information visualisation studies is to report: who has done what, the protocols followed for example during interviews, the number of participants and presenting the result in rigour. [56]

In “Preliminary Usability Evaluation of PolyMeCo: A Visualization Based Tool for Mesh Analysis and Comparison” by Santos et al. [57] a usability evaluation of the PolyMeCo tool is performed. The PolyMeCo tool is an integrated environment for mesh analysis. In the study they performed a heuristic evaluation with three persons with knowledge of usability to get a list with usability problems. They also held informal and formal evaluation sessions with students, though the students were not the intended users of the system. In the informal evaluation sessions the students were allowed to use the tool freely and afterwards they gave suggestions on how to improve the usability. In the formal evaluation sessions they had students perform predefined tasks while observing time, task completeness, fill a questionnaire and “other relevant information”. The study also had a different student group that had a week to freely use and evaluate the tool and get back with comments. [57]

In “Multiple Usability Evaluations of a Program Animation Tool” by Pérez-Carrasco et al. [58] an educational visualisation tool of software recursions was evaluated. This evaluation was done with the objective of evaluating the easiness of use, the adequacy to the task, the quality and if the user liked it. The students in the study got a brief demonstration of the tool then a simple task to familiarise themselves with the tool and lastly a larger assignment followed by a questionnaire with both questions using a Likert scale, described in section 4.2.4 and open questions. [58]
4 Method

In the following chapter the method and methodologies used in this study will be presented. First the overall structure of the execution of the study will be presented followed by a section containing the theories that this study is based on. After the theoretical parts of the method, the study’s method model will be presented with detailed sections for each part of the study. The different parts of the study are the following: an initial phase containing the background and planning and the choice of tool, a theoretical study, a pre-study containing interviews and a technical evaluation of the chosen tool, two iteration containing an implementation phases for implementing improvements in the tool and an evaluation phase for evaluating the tool. The chapter will be concluded with a section where the execution of the study is discussed from a critical perspective.

4.1 Study structure

Since this study contains several parts with different objectives, it was divided into several phases. The study had the following phases: an initial phase, a theoretical study, a pre-study and an implementation phase with two iterations that each ended with an evaluation.

The initial phase was divided into two parts. The first was to get better knowledge of Attentec and the second was an investigation on prototyping tools for aiding in development of data visualisations. From this investigation a decision on which tool that was the most suitable to perform further studies on was made. After the initial phase a theoretical study of visualisation, IoT and usability was conducted to form the theoretical framework. This theoretical study was done partly in parallel with the pre-study to be able to take both the practical and theoretical knowledge into account. The pre-study contained interviews with consultants to collect empirical data on how visualisations and IoT projects had been implemented previously at Attentec. A technical investigation was also conducted in this phase on the chosen prototyping tool to answer if there were any limits in using the tool for some characteristics or formats of data. The implementation and evaluation phases were done in two iterations. Based on the results from the interviews and the technical evaluation in the pre-study some initial modifications of the tool were made. The tool was evaluated in a usability study with the intended users of the system. The results were then evaluated and new improvements were made to the chosen tool based on the result to improve the usability. A
4.2 Method and methodologies

In the following subsections the theories that this study is based on will be presented. There are no universal rules how to perform a scientific study, instead it is the study’s research question that dictates what methods are suitable [59]. In the first three subsections the more philosophical values on which this study builds on will be described, these sections are the following: the scientific approach, scientific reasoning and research methodologies. Theories regarding questionnaires, interviews, literature, triangulation and controlled experiments will then be presented as the theoretical base for the moments performed in the study.

4.2.1 Scientific Approach

A few common scientific approaches are exploratory, descriptive, explanatory and normative. Exploratory studies are used when the prior knowledge in the area is small and the study tries to find a basic understanding in the area. Descriptive studies are used when there exist basic knowledge in the area and the goal is to describe but not to explain. Explanatory studies are used when searching for deeper understanding in an area and the goal is to both describe and explain. Normative studies are used when there exist some knowledge and understanding in the area and the goal is to give guidance and propose actions. [59] Since this study’s aim is to examine and give recommendations on how to develop visualisations of data using the chosen tool it is a normative study.

4.2.2 Scientific Reasoning

Depending on if a study has its origin in theory or in empirical evidence there are different scientific reasonings. If a study has its origins in theory and then tries to apply it to reality it is a deduction study. On the other hand if a study has its origins in theory and tries to find patterns to construct theories it is called induction. The third and final way is called abductive reasoning were moving between theory and empirical evidence is necessary to create a gradually emerging understanding. [59]

This study is mainly deductive. The study does however, to a certain extent, mix theory and empirical evidence since the visualisation tool was evaluated on features both from theory and from interviews with consultants. This empirical investigation of the tool was also used together with theoretical data to improve the tool. The empirical investigation was also the base for the user evaluations that yielded empirical evidence which, together with the theory, answers the research questions. Even if this study have some abductive elements the main goal is to take theories about visualisations and usability and then apply them to reality why this is a deductive study.

4.2.3 Research Methodologies

There are two different research methodologies, quantitative and qualitative which one that is most suitable to use depends on the objective of the study. Quantitative studies are used when the information is measurable. Qualitative studies are used when the information cannot be measured or consists of soft values. [59]

This study is mostly qualitative with some quantitative elements since the results are a mix of quantitative and qualitative. In the study quantitative scales are used to try to measure the qualitative parts since quantitative measurements are easier to analyse. This does not make the study quantitative rather a qualitative one that employs some quantitative methods.
4.2.4 Questionnaires

When it comes to questionnaires both Jamieson and Ejvegård mention the importance of having clear questions that leaves none or very little room for interpretation [60, 61]. Likert scales are often used as a way of measuring a feeling or attitude. The scale provides a range of responses to a question or a statement. There are usually five categories of response, from 1 strongly disagree to 5 strongly agree. There is research that argues that scales with seven or with an even number of response categories is better than using five categories. [60]

Since scales with five alternatives are most commonly used, a Likert scale with five alternatives will be used in this report. Ejvegård proposes that the scales always shall have an odd number of categories and that the category in the middle can be ignored when summarising the responses since choosing this option shows signs of insecurity [61].

4.2.5 Interviews

Interviews and observations are both primary resources which means that the data is collected specifically for the specific study [59]. There are three main types of interviews: structured, semi-structured and unstructured.

Structured interviews are when the questions and the order of the questions are fixed. Semi-structured interviews are when the topics are decided but not all questions so that the person conducting the interview can ask follow up questions depending on the respondents answers. Unstructured interviews are interviews where the questions are not decided before the interview so that the interview more reassembles a dialogue. [59]

Recording interviews should only be done with the consent of the respondent. The benefits with recording an interview are that the interviewer does not have to take notes and that it is possible to listen to the recording again afterwards. There are also some drawbacks with recording an interview. Transcription of the interviews are often required which can take up to four or five times the length of the interview itself. The presence of a recording device could also affect the respondents responses negatively and inhibit spontaneous answers. [62]

When conducting interviews, there are some important things to take into account. It is important to have a clear intention on what the interview should provide in terms of acquired knowledge and what the information should be used for. Questions that only has a short yes or no answer should be avoided as well as asking multiple questions at the same time since this makes it hard to respond. The questions used should also be simple to understand without any implications, presumptions or loaded words. The interviews should also be well planned and the interviewer should be prepared, keep to the subject and have prepared possible follow up questions. [63]

4.2.6 Literature

When using literature as a source the information has been collected by someone else that might have had other purposes than this study. They might therefore have interpreted the information to fit that purpose. This is called secondary sources, data that has been collected in other studies. The data they present can be angled to their purpose or incomplete since they might have had a different purpose. It is therefore important to use data from secondary sources with caution and when ever possible use multiple sources. [59]

4.2.7 Triangulation

To verify the data collected for this study multiple sources have been used for both the theoretical parts and in the empirical parts. This is called triangulation and is done to increase the credibility and validity of the results by showing more than one persons opinion. Triangulation is when instead of using one source, multiple sources are used to get different opinions or to show consensus on the subject [59, 64]. The main areas where triangulation
has been used in this study is the theoretical framework, interviews in the pre-study and the evaluations.

4.2.8 Controlled experiments

In “A practical guide to controlled experiments of software engineering tools with human participants” Ko et al. [65] discusses experiments in software development using humans. In short the process has the following steps.

**Recruitment**, the process of advertising the study.
**Selection**, the selection of participants using an inclusion criteria.
**Consent**, the approval from the participant to take part in the study.
**Procedure**, the information about what the participant shall do in the study.
**Demographic measurements**, information about the demographics of the participants.
**Group assignment**, assignment of participants to groups, for example a reference group.
**Training**, teaching the participant how to use the task before the actual study.
**Tasks**, tasks to perform with the tool.
**Outcome measurements**, measurements and results from the tasks performed.
**Debrief and compensate**, explain for the participant the purpose of the study and compensate for the time. [65]

These steps were used as a basis for the evaluations in this study. A more in depth description of how the steps were performed are described in section 4.3.5 and 4.3.7.

4.3 Method Model

As mentioned earlier, this study consisted of two initial phases followed by an implementation and evaluation phase with two iterations. A graphical overview of the phases can be seen in figure 4.1.

---

**Figure 4.1: Method model.**

First an initial phase was conducted to determine the objective of the study containing discussions with the company and the university and the creation of the thesis plan. During the interviews with the company a list of requirements that they would like for a prototyping
tool to have was compiled. With that compiled list as a starting point a comparison of different prototyping tools for aiding in development of data visualisations were made. From this comparison the tool that was the most suitable to Attentec's needs was chosen to perform further studies.

This initial phase was followed by a theoretical study to investigate different ways of visualising data and what problems that can occur when large amounts of data needs to be visualised. This study examined IoT, big data, various approaches and visualisation classifications and visualisation tools. One delimitation in this part was that only those in the literature most common methods were to be studied further. The literature review on the IoT and big data subject also addressed structures and characteristics in data from IoT systems that can influence the visualisation and usability in software.

Partly in parallel with the theoretical study a pre-study was conducted that contained interviews with consultants at Attentec and a technical investigation of the chosen tool. The interviews had the objective of collecting empirical data on how visualisation projects are conducted and the technical investigation had the objective to investigate if there are limits in using the chosen tool for some characteristics or formats of data. The tool was examined using open sensor data from the Swedish Meteorological and Hydrological Institute, SMHI, and from the Swedish Transport Administrations (Swedish: Trafikverket) open APIs. The data from these sources were chosen as it was quite similar to IoT data. The tool was examined by using the knowledge gained from interviewing consultants at Attentec about earlier IoT and visualisation projects and the characteristics identified in the theoretical framework.

After these phases, an implementation phase followed by an evaluation phase, with collection of empirical data, was conducted. This implementation phase and evaluation phase were executed twice in so called iterations. From the pre-study, initial modifications to the tool were made based on the results from the interviews and technical evaluation in the pre-study. A hypothesis was produced from the results of the pre-study and the research question. The hypothesis in the first iteration was that the chosen tool with the initial modifications is usable from a developer’s perspective about the tool’s perceived usability and that it would assist in the development of a visualisation of data. The hypothesis in the second iteration was that the chosen tool with the modification made in the implementation phase improved the usability from a developer’s perspective about the tool’s perceived usability and that it would assist in the development of a visualisation of data. To test the hypothesis a few small tasks were produced that the participants of the evaluation were to perform with the tool. These tasks were designed to test the tool and the usability of it. After the tasks were completed the participant was given a short survey and was interviewed about their experience. This was done to answer the following question: “Is the chosen tool for the case study usable from a developer’s perspective on the tool’s perceived usability”.

4.3.1 Initial phase

A small initial phase was performed first. This was phase split into two parts. In the first part the goal was to get a better knowledge of Attentec, their situation and if there had been any prior work on the subject to be able to make the objective of this study clear. From this part a list with criteria that Attentec would like for a prototyping tool to have was compiled. The second part was an investigation on prototyping tools for aiding in development of data visualisations and their limits and benefits. The tools that were found were then compared to the requirements that Attentec would like the tool to fulfil. From this comparison a decision on which tool that was the most suitable to perform further studies on was made.

Initial phase, part 1

The goal with the first parts of this phase was to make the objective clear and to derive a list of requirements that Attentec had on a prototyping tool for data visualisations. In this phase
a discussion was conducted with representatives from Attentec. Their desires were then used and taken in to account to clarify the objective for this study. The objective of the study was then presented to the tutor at Linköping University to ensure that the study had enough depth. This process continued iteratively until all parts were satisfied with the objective of the study. The discussions with consultants from Attentec also yielded a list with criteria that they would like a prototyping tool for data visualisations to conform to.

**Initial phase, part 2**

In the second part the goal was to choose the most suitable prototyping tool for aiding in the development of visualisations of IoT data. For this the compiled list of criteria from Attentec was used when selecting and comparing different tools.

The tools were found by using Google and searching for the keywords presented in the list below. From the search results the 5 highest non sponsored links were explored and the tools mentioned were added to the list. Software databases were excluded from the search hits.

Search keywords:

- visualization tool
- data analytics software

If any tools that were mentioned during the interviews in the initial phase was not on the list after this search they were added manually to the list. This compiled list of visualisation tools was then compared with the requirements that Attentec would like the tool to have. Tools matching the constraints were examined further, by either installing and testing or by reading about the tool. The tool that was the most versatile yet simple was then chosen as the tool for further evaluation.

**4.3.2 Theoretical Study**

The goal of the theoretical study was to investigate relevant theories about visualisation, usability and data evaluation. This was done through a literature study where both books and articles have been consulted.

Since this phase only focuses on literature, which is a secondary source as mentioned in the literature section in 4.2.6 caution has been taken when reading and referencing to these sources. One way of doing this and at the same time improving the credibility of the literature study is to use triangulation, as mentioned in 4.2.7. In this study triangulation has been used in the main parts of the theoretical framework.

Sources in the theoretical framework are mainly peer-reviewed research articles which have the advantage of often containing newer technology and theories compared to those found in books. The books used are often older but creates a solid base of accepted theories that newer theories can be built upon.

**4.3.3 Pre-study**

In the pre-study the main goal was to gain knowledge about how Attentec works with visualisation and IoT-systems today and to do a technical investigation of the chosen tool.

**Interviews**

Interviews with consultants that have experience in implementing data visualisation were held to identify the work process, tools and what characterises a data visualisation. There
were also interviews held with consultants that had worked on IoT-projects to gain knowledge about the work process, the customer involvement and if there are any problems commonly occurring during projects. Consultant managers with insight in multiple projects that the company has worked on were also interviewed to gain knowledge about the work process, customer involvement and if there are any common or important characteristics from the different IoT projects.

All interviews held in this phase were semi-structured, as described in the Interview section of 4.2.5. Since there were some knowledge about the area beforehand, semi structured interviews were deemed as a suitable technique to use. By using semi-structured interviews instead of structured ones follow up questions could be used to clarify or investigate an answer in more detail. When conducting the interviews, triangulation techniques were employed so that several persons were interviewed on the same topics to bring forward different opinions.

The guidelines presented in the Interview section of 4.2.5 were used to develop the interview guides and during the interviews. All interviews were held in Swedish and the interviews where recorded and anonymised. The benefits of recording the interviews were deemed to outweigh the drawbacks, especially since the interviews were performed by a single interviewer so the risk of not being able to take notes at the required speed seemed too big. Anonymisation of the interviews together with the topic not being that sensitive were also factors that affected the choice of recording the interviews.

The interviews were performed in person when it was possible. When not possible to meet in person due to for example not being stationed in the same office the interview was performed with Skype. The respondents received the questions through email a few days before the interview to be prepared on which questions that would be asked. This was done as an attempt to keep the interviews shorter and on topic by letting them prepare before the interview.

During this phase six interviews were conducted, three with consultants and three with managers. The interviews had two parts and depending on the interviewed person’s background questions regarding IoT and work method were asked and/or questions about data visualisation. In table 4.1 information about the interviews is presented. The table shows in which of the two areas that each person were questioned and which interview guide that was used. The interview guides used can be found in the appendices indicated in the table.

<table>
<thead>
<tr>
<th>Person</th>
<th>IoT and work method</th>
<th>Visualisation</th>
<th>Interview guide</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consultant 1</td>
<td>X</td>
<td>X</td>
<td>Pre-study regarding visualisation, with consultant A.1</td>
</tr>
<tr>
<td>Consultant 2</td>
<td>X</td>
<td></td>
<td>Pre-study regarding IoT and work method, with consultant A.3</td>
</tr>
<tr>
<td>Consultant 3</td>
<td>X</td>
<td>X</td>
<td>Pre-study regarding IoT, visualisations and work method, with consultant A.4</td>
</tr>
<tr>
<td>Manager 1</td>
<td>X</td>
<td></td>
<td>Pre-study regarding IoT and work method, with manager A.2</td>
</tr>
<tr>
<td>Manager 2</td>
<td>X</td>
<td></td>
<td>Pre-study regarding IoT and work method, with manager A.2</td>
</tr>
<tr>
<td>Manager 3</td>
<td>X</td>
<td></td>
<td>Pre-study regarding IoT and work method, with manager A.2</td>
</tr>
</tbody>
</table>

Table 4.1: Interviews during pre-study.
4.3. Method Model

Technical investigation

A technical investigation of the chosen tool was performed to see if there were any limits in using the tool for some visualisations, characteristics or formats of data. The characteristics from the theoretical framework together with information from the interviews were compiled into a checklist and examined through implementations with the chosen tool. The tool was examined in a technical setting using open sensor data from SMHI and from Trafikverkets open APIs as representative data for IoT, to test some limitations that were not obvious. Selected data from both sources were also locally stored in a MySQL and a MongoDB database to examine if it was possible to gather data from those sources too. The following data sets from the previously mentioned APIs were used during this evaluation.

- Weather station data from all of Trafikverkets weather stations from the latest hour.
- Weather station data from all of Trafikverkets weather stations collected during one week using a sample frequency of 30 minutes stored in a MongoDB database.
- Temperature data from SMHI measurement station in Norrköping from the last 20 years.
- Temperature data from SMHI measurement station in Norrköping from the last 20 years stored in a MySQL database.

The evaluation was performed by implementing small demonstrations or simply by looking if the support existed in the interface. The checklist used in this evaluation is found in appendix B.1.

4.3.4 Implementation, iteration 1

In the first implementation phase the functionality of the chosen tool was increased. This functionality was derived from the limits discovered in the chosen tool in earlier phases as well as from the interviews held with the consultants in the earlier phases.

The functionality was added in the form of a visualisation plug-in. The plug-in was developed using Ottersten and Berndtsson thoughts about the intended users needs and Shneiderman and Plaisant Eight Golden Rules of interface design, both described in section 2.6.1. By having these guidelines in focus throughout the design process the plug-in was developed with usability as the primary focus.

From this the hypothesis was that the chosen tool with the initial modifications is usable from a developer’s perspective about tools perceived usability and that it would assist in the development of a visualisation of data.

4.3.5 Evaluation, iteration 1

The process of evaluating the tool followed the steps from Ko et al. [65] in section 4.2.8.

The recruitment and selection of participants were made together with one of the managers at Attentec to get three consultants that was stationed at the office in Linköping. Two master thesis students were also asked to participate in the evaluation since the only criteria for the selection were that they should be at Attentec’s office in Linköping and that they should have programming knowledge and be familiar with SQL syntax. The participants were informed about the study and gave their consent to participate in the evaluation. No Demographic measurements or Group assignments were performed due to them not being relevant to this evaluation.

To evaluate the tool eight tasks were produced to be conducted with the tool. The tasks were based on the information on IoT characteristics from the interviews and the technical investigation in the pre-study. The tasks were divided in two parts. The first part was to
import the data to be used and the second part were to create some visualisations of the data. Both parts were designed using the result of the pre-study interviews to create realistic tasks as if prototyping visualisations to a customer.

The data used in this task was chosen based on the results from the pre-study so that it would be representative for data in an IoT project. The visualisation tasks were created by playing with the data to generate visualisations that highlighted some interesting feature. From that visualisation a task was written that should generate the visualisation.

The tasks and the evaluation questions described later in this section were designed with the results from the pre-study and the following research question in mind: “Is the chosen tool for the case study usable from a developer’s perspective on the tools perceived usability”.

Before the evaluation a short, only a couple of minutes long, introductory interview was held with each participant. The purpose with these interviews was to check the participant’s knowledge level and previous knowledge of the subject to confirm that they matched the selection criteria. The interview guide for these interviews can be found in appendix A.5.

After the introductory interview a demonstration of the tool for about 3-5 minutes were given for the participant. This was done to give a brief overview of the tool and to let the participant ask any question before the evaluation started.

The participants were given a new project with text sections containing instructions to the tasks together with information about where the data were stored. An example of a data record was also given to the participant, this can be seen in appendix C.1. The participant was instructed to read the instruction sheet carefully and ask any questions if something was unclear for a maximum of 5 minutes. No extra information was given as answer to those questions, only clarifications about the tasks’ formulation.

When the five minutes were up the time started and the participant had to perform as many of the tasks as possible or all tasks during 60 minutes. If the task of importing the data exceeded 30 minutes the test leader provided a working data import script and noted this so that the participant could continue with the other tasks. If the participant got stuck on a task the participant could choose to skip the task and continue with the next task. Some examples, shown in the demonstration of the tool, containing solutions to similar tasks were available during the whole time for the participant as a reference and a way to limit the participants programming skills as a variable in the evaluation. Having examples similar to the tasks were also reasonable since if using the tool in a real world setting it is likely that similar solutions from earlier projects would exist as references.

After each sub-task was completed a questionnaire with three questions was filled out by the participant, this questionnaire can be found in appendix D.1. This questionnaire was used to measure the user’s satisfaction with the tool for the different tasks. During the evaluation the time to perform a task and task completion were tracked by the test leader. Task satisfaction, task time and task completion were then used to calculate a modified SUM value where the error rate was omitted. The error rate was left out since it was hard to define what an error was in this context. There was also thought to be a risk that the participants would feel inhibited and afraid to explore the software if errors were counted.

The SUM value were calculated as described in section 2.6.2 about summed usability metric but without the errors. 4.0 was used as the specification limit for satisfaction as recommended by Sauro and Kindlund [49]. The specification limit for the time for each task was set by the author doing the tasks after having used the tool and gaining experience in the tool. The time it took with experience of the tool multiplied by 1.5 was set as the specification limit for each task.

After a participant had completed all the tasks or the allotted time was up a short survey containing the SUS questionnaire, found in appendix E.1, was filled out by the participant. An interview was also conducted to establish how satisfied the user was with the tool and what the user saw as the biggest opportunity for improvement. The interview guide used
for these interviews can be found in Appendix A.6. The interviews were semi-structured as described in 4.2.5 so that the respondent could answer freely and follow up questions could be asked. The evaluation ended with thanking the participant for their time and contribution.

The results of the interviews and questionnaire were analysed and improvements proposed where the respondents had found flaws in the usability.

### 4.3.6 Implementation, iteration 2

The proposed improvements from the first evaluation were ordered and grouped depending on the number of participants that mentioned them and smaller improvements on the same part were grouped together.

The improvements that were mentioned by the most participants were then developed. The improvements were designed by using Östersten and Berntsson thoughts about the intended users needs, which were identified in the pre-study, and Shneiderman and Plaisant Eight Golden Rules of interface design both described in section 2.6.1. These guidelines were in focus throughout the development process so that the developed plug-in was developed with usability as the primary focus.

The hypothesis in the second iteration was that the chosen tool with the modification made in the implementation phase improved the usability from a developer’s perspective about tools perceived usability and that it would assist in the development of a visualisation of data.

### 4.3.7 Evaluation, iteration 2

The evaluation was performed in the same manner as described in Evaluation, iteration 1 in section 4.3.5 with the following modifications.

No new recruitment of participants were conducted, instead the same persons as in the first evaluation were used for the second evaluation as well. The main reason for this was that by using the same participants in both evaluations any difference in the SUS and SUM scores were not caused by new participants with different opinions than the first group of evaluators. Since all participants also had the same amount of training in the tool, it did not affect any comparison between the participants. The learning curve of the persons were deemed to be insignificant since the only usage of the tool they have had were the hour in the previous iteration.

New tasks were created that can be seen in Appendix C.2. The tasks have the same general structure and complexity as the tasks from the first evaluation to be able to compare the results.

The SUM metric and SUS metric were compared against the results from the previous iteration to determine if the usability had been increased.

### 4.4 Method criticism

In this section, the method used in this study will be discussed from a critical perspective to give the reader a chance to personally decide upon the reliability of the methods used and through this increase the study’s credibility.

The three concepts of reliability, validity and objectivity together form the study’s credibility and should be as high as possible. Reliability is how reliable the results are. Validity is how well a study performs on measuring the objective of the study. Objectivity is the way in which the author’s values impact the study.

The choice of doing a qualitative study has a weakness in that qualitative studies are more difficult to generalise. To make sure the study have a high validity even though it includes mostly qualitative aspects, quantitative parameters have been used when possible to quantify the otherwise qualitative values. This also increases the objectivity of the study.
since the risk that the authors view influenced the outcome are reduced when using quantitative parameters.

The literature study has the disadvantage that it is secondary data which means it is initially collected for a different purpose than this study has [59]. To minimise the influence of this triangulation has been used which means that multiple sources and perspectives are taken into account to give a complete picture [59]. For reliability to be as high as possible the major parts on which the report is based has used triangulation.

Since Internet of Things is an area where intense research is being made, articles have been the primary source of literature. This is because of the fact that articles often contains information that is more up to date than books. For the same reason mostly articles published in recent years has been used since these are the ones that contains the most relevant information. For the other parts of the study older articles have also been consulted. Usability for example is a more mature subject where old articles might contain valuable methods to measure it.

Other methods that could have been used is for example to perform a broader study to look on multiple companies and solutions and examine those. This would require companies to help out in a larger extent and was there for deemed unsuitable for this study.

Instead of semi-structured interviews, structured interviews could have been performed. This would have made the interviews easier to perform but could have resulted in not getting the interesting information about the subject.
In this chapter the results of the study are presented. The chapter is structured so that each phase has a section describing the results from the specific phase and lastly a concluding section where the results are compiled.

5.1 Initial phase

In this section the results from the initial phase will be presented. The phase was split into two parts, the first regarding the background and planning of the study but also the requirements of a visualisation tool and the second part contained the selection of the visualisation tool to be evaluated as a tool for prototyping data visualisations.

5.1.1 Background and planning

The result from this part was a thesis plan containing a preliminary title, a problem description, preliminary approach, a literature base, a list of relevant courses, a time plan, and a list of risks and a contingency and mitigation plan for the risks. None of the above mentioned results hold any significant scientific value that needs to be presented here.

From the discussions with Attentec a list of requirements that they would like for a visualisation tool to fulfil was derived. For this study the following constraints were therefore deemed to be important when choosing the visualisation tool:

- The tool had to work on Linux, Windows and Mac since customers have different needs and requirements.
- It had to be able to run on a local machine so that no data can leak or be viewed by unauthorized persons.
- It had to be free to use for companies.
- The tool had to be open source to enable modifications or be open for plug-in development.
- The tool had to have some built in charts or data analytics that can be used out of the box.
• The tool had to have more than one type of charts built in.

5.1.2 Visualisation tools

There are numerous tools for data visualisation that can be used for prototyping. To find as many suitable tools as possible they were found using Google and searching for the keywords presented below. From the search results the 5 highest non sponsored links were explored and the tools mentioned were added to the list. Software databases were excluded from the search hits.

Search keywords:
• visualization tool
• data analytics software

If any tools that were mentioned during the interviews in the initial phase was not on the list after this search they were added manually to the list.

The result of this search with the above mentioned parameters can be seen in alphabetic order table [5.1]. The search string that the tool were found with is indicated in the column to the right “Search string” together with the source. In this table all the found tools are displayed even if it did not fulfil all the requirements. Graph libraries and programming languages were excluded since they are not a prototyping tool according to Attentecs definition.

In the cases where a requirement excluded a tool from further investigation this is indicated in the middle column. It is always the first requirement found from the list to not be fulfilled by the tool that are listed in the middle column “Reason for exclusion”. There might be cases where several requirements are not fulfilled but since the investigation stopped as soon as one requirement was not met this is not researched in this study.

<table>
<thead>
<tr>
<th>Tool</th>
<th>Reason for exclusion</th>
<th>Search string</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Zeppelin</td>
<td>Mentioned during the initial phase.</td>
<td></td>
</tr>
<tr>
<td>CartoDB</td>
<td>Only online</td>
<td>“visualization tool” [66]</td>
</tr>
<tr>
<td>Candela</td>
<td>JavaScript library</td>
<td>“visualization tool” [67]</td>
</tr>
<tr>
<td>Chart.js</td>
<td>JavaScript library</td>
<td>“visualization tool” [68, 66]</td>
</tr>
<tr>
<td>ChartBlocks</td>
<td>Only online</td>
<td>“visualization tool” [68, 67]</td>
</tr>
<tr>
<td>ChartBuilder</td>
<td>“visualization tool” [69]</td>
<td></td>
</tr>
<tr>
<td>Charted</td>
<td>“visualization tool” [67]</td>
<td></td>
</tr>
<tr>
<td>Chartio</td>
<td>Not free</td>
<td>“visualization tool” [70]</td>
</tr>
<tr>
<td>Chartist.js</td>
<td>JavaScript library</td>
<td>“visualization tool” [67]</td>
</tr>
<tr>
<td>Clearify Qube</td>
<td>Not free</td>
<td>“data analytics software” [71]</td>
</tr>
<tr>
<td>Crossfilter</td>
<td>JavaScript library</td>
<td>“visualization tool” [66]</td>
</tr>
<tr>
<td>D3.js</td>
<td>JavaScript library</td>
<td>“visualization tool” [66, 68, 67]</td>
</tr>
<tr>
<td>DataHero</td>
<td>Online online</td>
<td>“visualization tool” [66]</td>
</tr>
<tr>
<td>Dataiku DSS</td>
<td>“data analytics software” [72]</td>
<td></td>
</tr>
<tr>
<td>DataMelt</td>
<td>Graph library</td>
<td>“data analytics software” [72]</td>
</tr>
<tr>
<td>Datawrapper</td>
<td>Only online</td>
<td>“visualization tool” [68, 67]</td>
</tr>
<tr>
<td>DevInfo</td>
<td>Only online</td>
<td>“data analytics software” [72]</td>
</tr>
<tr>
<td>Domo</td>
<td>Only online</td>
<td>“visualization tool”, “data analytics software” [70, 71]</td>
</tr>
<tr>
<td>Dygraphs</td>
<td>JavaScript library</td>
<td>“visualization tool” [66, 68]</td>
</tr>
<tr>
<td>Excel</td>
<td>Not free</td>
<td>“visualization tool” [66]</td>
</tr>
<tr>
<td>Exhibit</td>
<td>JavaScript library</td>
<td>“visualization tool” [66]</td>
</tr>
<tr>
<td>Tool</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>Flot</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Fluentd</td>
<td>No built in graphs</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Fusion Charts</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Fusion Tables</td>
<td>Only online</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Gephi</td>
<td>Only network graphs</td>
<td>“visualization tool”, “data analytics software”</td>
</tr>
<tr>
<td>Google Charts</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Grafana</td>
<td>Only time series</td>
<td>Mentioned during the initial phase.</td>
</tr>
<tr>
<td>Graphite</td>
<td>No Windows support</td>
<td>Mentioned during the initial phase.</td>
</tr>
<tr>
<td>Highcharts</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>iCharts</td>
<td>Only online</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>ILNumerics.NET</td>
<td>Graph library</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Infogram</td>
<td>Only online</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>InstantAtlas</td>
<td>Not free</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Intelex</td>
<td>Not free</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Import.io</td>
<td>Not free</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>IPython</td>
<td>No built in graphs</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Italassi</td>
<td>Math graphs only</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>JavaScript InfoVis Toolkit</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>jqPlot</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Julia</td>
<td>Programming language</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Kartograph</td>
<td>Only geospatial data</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Kibana</td>
<td></td>
<td>Mentioned during the initial phase.</td>
</tr>
<tr>
<td>KNIME Analytics platform</td>
<td></td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Leaflet</td>
<td>Only geospatial data</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Logi Info</td>
<td>Not free</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Looker</td>
<td>Only online</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Matplotlib</td>
<td>Graph library</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Microsoft Power BI</td>
<td>Not free</td>
<td>“visualization tool”, “data analytics software”</td>
</tr>
<tr>
<td>Modest Maps</td>
<td>Only geospatial data</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Mozenda</td>
<td>Not free</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>MyHeatMap</td>
<td>Only online</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>NodeBox</td>
<td></td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>NodeXL Basic</td>
<td>Only network graphs</td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Openheatmap</td>
<td>Only geospatial data</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>OpenLayers</td>
<td>Only geospatial data</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>OpenRefine</td>
<td>No built in graphs</td>
<td>“visualization tool”, “data analytics software”</td>
</tr>
<tr>
<td>Orange</td>
<td></td>
<td>“data analytics software”</td>
</tr>
<tr>
<td>Palladio</td>
<td></td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Plotly</td>
<td>Not free</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Polymaps</td>
<td>Only geospatial data</td>
<td>“visualization tool”</td>
</tr>
<tr>
<td>Qlik Sense Desktop</td>
<td>Only works on Windows</td>
<td>“visualization tool”, “data analytics software”</td>
</tr>
</tbody>
</table>

5.1. Initial phase
<table>
<thead>
<tr>
<th>Tool</th>
<th>Availability</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QlikView</td>
<td>Not free</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Programming language</td>
<td>“visualization tool”, “data analytics software”</td>
<td></td>
</tr>
<tr>
<td>RapidMiner</td>
<td>Not free</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Raw</td>
<td></td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>ROOT</td>
<td>Graph library</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>SAP Lumira</td>
<td>Not free</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>SAP Roambi</td>
<td>Not free</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>SAS</td>
<td>Not free</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Silk</td>
<td>Only online</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Sisense</td>
<td>Not free</td>
<td>“data analytics software”, “data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Spotfire</td>
<td>Not free</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Style Scope</td>
<td>Not free</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Tableau Desktop</td>
<td>Not free</td>
<td>“visualization tool”, “data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Tableau Public</td>
<td>Only online</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Tanagra</td>
<td>Only ScatterPlot</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Tangle</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Terapeak</td>
<td>Not free</td>
<td>“data analytics software”</td>
<td></td>
</tr>
<tr>
<td>Timeline</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Visualize Free</td>
<td>Only online</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Weave</td>
<td></td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>WebFOCUS</td>
<td>Not free</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Weka</td>
<td>Mainly a tool for machine learning</td>
<td>“visualization tool”, “data analytics software”</td>
<td></td>
</tr>
<tr>
<td>WolframAlpha</td>
<td>Only online</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>ZingChart</td>
<td>JavaScript library</td>
<td>“visualization tool”</td>
<td></td>
</tr>
<tr>
<td>Zoho reports</td>
<td>Only online</td>
<td>“visualization tool”, “data analytics software”</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1: Search hits.

Of the 88 tools that were identified as suitable for creating visualisations only 11 tools fulfilled all the requirements. The tools that met the initial requirements are presented below with some information about each tool.

**Apache Zeppelin**, an open source interactive tool where the user writes code to collect and modify data. The data can then be queried using for example SQL and visualised using built-in or third-party graphs [76].

**ChartBuilder**, a light weight tool for generating charts limited on both what kind of graphs are possible and which data formats that can be used. [77]

**Charted**, also a light weight tool with limits on what kind of charts that can be used and what can of data formats that can be used. [78]

**Kibana**, a visualisation tools that supports many different kinds of charts but requires the data to be stored in Elasticsearch. [79] This limits its use cases or requires the data to be stored in several places.

**KNIME Analytics platform**, uses a drag and drop interface for programming the flow with the option to add custom scripts. [80]

**NodeBox**, resembles KNIME in that it uses a visual drag and drop programming interface for the flow. Nodebox is smaller though and can only import data from CSV files in the
5.1. Initial phase

Orange, which has a visual programming interface and also some built in visualisations. The tool comes with the possibility to import data from Excel, CSV, ISV, SQL or by scripting in Python.

Palladio, has built in visualisations and can load CSV and ISV files.

Raw, a drag and drop tool where the user drags a data file in to the tool and then chooses the graph type before selecting axes and setting. The tool then generates a code snippet that can be included on a web page. The tool lacks support for dynamic data and storing the graphs require additional steps.

Weave, has built in visualisations and supports CSV, GeoJSON, SHP/DBF and CKAN files.

ChartBuilder, Charted, Palladio and Raw were not chosen since they were small tools limited to CSV files and a few graphs. KNIME, NodeBox and Orange were not chosen since they build on a visual drag and drop interface for programming. This is a feature that could be seen as a positive trait but it limits the developers to only use the blocks provided or write their own blocks. It also requires the users to learn the syntax of the drag and drop interface which is why they where not chosen. Kibana was not chosen since it requires the data to be stored in Elasticsearch. Weave also have limitations on what data sources that can be used and was therefore not chosen.

After these rejections only one tool remained, Apache Zeppelin. Therefore, from the selection and comparison of the tools, Apache Zeppelin, hereinafter referred to as Zeppelin, was chosen to be evaluated as a tool for prototyping visualisations. Zeppelin was chosen since it was not limited to any specific graph types or storage types and did not require additional tools. Zeppelin requires the user to know programming to load the data but does not limit the import of data to a specific data source or format and leaves the programmer free to use a familiar programming language.

The first sub question to the research question “Which tool is most suitable for using as a prototyping tool for data visualisations regarding Attentecs needs?” can now be answered. Using the criteria and looking at the tools that passed the criteria in this study, Apache Zeppelin is the most suitable tool for prototyping data visualisations based on information presented about the tools.

5.1.3 The chosen tool

Zeppelin is an open source interactive tool for data analytics and data visualisation. It has a web-based notebook interface with four main parts according to the documentation: Data Ingestion, Data Discovery, Data Analytics and Data Visualization & Collaboration.

It is possible to program languages such as Apache Spark, Python, R, JDBC, Markdown and Shell and render HTML and JavaScript. Zeppelin has an Apache Spark integration that provides SparkContext and SQLContext injection that can be used to load and query the data to visualise. Apache Zeppelin can dynamically create input forms in, for example, the SQL commands to filter data on user input.

Zeppelin comes with some basic charts: scatter plots, column charts, line charts and pie charts, where the user can drag and drop to use the different columns as axes or values. A package handler called Helium is also included in Zeppelin that can be used to install more charts and interpreters. As of the 28 of March 2017, Zeppelin version 0.8.0-SNAPSHOT, 6 additional charts where available as plug-ins: a heat map, two types of bubble charts, a scatter plot, a spline and a column range. Additionally, Helium has support for developing or installing third-party charts as well.

Since Zeppelin was deemed to be the most suitable tool for Attentecs needs it will be
5.2. Pre-study

Zeppelin can be argued to be both a low-fidelity prototyping tool and a high-fidelity prototyping tool. Low-fidelity since it is limited to visualisations and high-fidelity since it allows the developers and the customer to interact with the visualisations. It also has advantages and disadvantages from both high- and low-fidelity. The advantages and disadvantages from high- and low-fidelity prototyping that are applicable on Zeppelin are listed below.

The advantages from low-fidelity are a lower development cost, the prototype can evaluate multiple design concepts and it can be used as a proof-of-concept for visualisations. Most of the disadvantages from low-fidelity prototypes are mitigated since error checking occurs when the data is imported and visualised. The prototype might however have limited utility after the requirements are established.

The advantages from high-fidelity are that the prototype will serve as a living specification and as a marketing and sales tool. The prototype might not have complete functionality the fully interactive experience requested by the customer but will still have the core functionality and basic interactivity. The disadvantages for high-fidelity prototypes of being more time-consuming and expensive to develop is partly true for prototypes in Zeppelin since it takes more time to use than to draw on paper but not as much time as to develop a complete high-fidelity data visualisation from scratch. Zeppelin could also have the disadvantage of the customer focusing on other parts of the prototype than the visualisation. Management could also think it is a real product if it is not clearly communicated that it is only the visualisations that is what is being prototyped.

5.2 Pre-study

In this section the results of the interviews and the technical evaluation of Zeppelin will be presented.

5.2.1 Interviews

The interviews were performed in person for all but two persons, consultant 2 and consultant 3. They were instead interviewed over Skype due to not being stationed in the office. In the following sections a summary of each interview will be given.

In summary consultant 1 had implemented a visualisation while being employed at a different company and was therefore only questioned about the visualisation not the IoT and work method. The visualisation was of blog data and the project was requested from a PR firm. The customer knew what they wanted at the start of the project so no prototype was made, instead the customer could point to similar visualisations.

The graph type used in the visualisation was a network graph, with different sized nodes to indicate size of the blog and overlays and colouring to show topics. The framework that was tested lacked support for what they wanted to do. This resulted in that they had to write the visualisation by them self using graphical libraries and the original framework for generating node positions. The data were stored in a SQL database but it was hard to access it for the task. This was due to the format and size of the data but also because it was sharded in the database. The biggest problem in the project was data management which included a lot of manual work.

The second consultant had worked on an IoT project that connected the product to a website to collect data and control some aspects of the product remotely. The project was developed in an agile manner. The customer was sitting in another town in the project which caused some communication issues. The project had a lot of changes in the requirement’s specification. They worked with prototypes to get feedback from the customer early on in
the process. The communication with the customer was mostly during sprint meetings. The idea was unclear from the beginning and they had to work out the requirements together with the customer during the project.

In the project Attentec was responsible for the software solution and another company for the hardware. The hardware company had their own solution with a special router and a REST endpoint that were causing problems during the development. The consultant interviewed felt that the solution missed a layer responsible for, among other things, security and scalability. All those problems led to some disappointment from the customer. The data were stored in a NoSQL database, MongoDB. The data contained position data which were used for geofencing, it also contained sensor data from the product. No visualisation was done, the project only contained displaying data and interaction with the product why consultant 2 was not interviewed regarding visualisations.

The third consultant worked on a project about visualisation of sensor data from sensors on one product. This was not an IoT project and the collection of data was handled by the customer. The project was developed in an agile manner. The project was to replace an old system and was built in Windows using .NET. In the project they used a graphical designer to create mock-ups and proposals on how the interface could look. The customer involvement in this project was large and status meetings with the customer were held every day.

One of the hard parts was that the system needed to be modular so that the users could show what they were interested in and that it should require little to no interaction from the user since the user had other things to do at the same time. The data was collected from an API, a WCF, Windows Communication Foundation. The data was not stored in the application so the data were only kept in memory, but the API allowed for collection of historical data from the last twelve hours. The hardest part was to understand the data since neither the consultants nor customer were the users of the system. Making the interface as simple as possible and modular was also one of the hardest parts. The visualisations in the system are meters to show for example fill ratio, symbols that show angles, textual information that show measurement data, radar chart that show multiple variables in one graph, histograms and line diagrams.

In the interview with manager 1 the following were mentioned. In the majority of the projects an agile project methodology is employed, using iterative development cycles. This is however dependent on the customer and if the customer have different requirements on the development methodology. There is nothing in general that differs between an average project and an IoT project in the project methodology that is employed. What does affect the project methodology is the customers technical knowledge level in the area. In most cases the requirement specification is developed together with the customer.

In the first iteration it sometimes happens that prototypes are produced and used, but it is also common to start on the real product without any prototypes. The customers’ interaction and involvement in the project is mostly dependent on their knowledge level and if they have any requirements or requests, and ranges from being on sprint demos to attending daily status meetings. The interaction between hardware and software could be one of the hardest parts in implementing an IoT system. Position data together with sensor readings are thought to be common in data from IoT systems. A trend towards larger data sets and big data were also mentioned.

In the interview with manager 2 the following were mentioned. The general work process is both ad hoc and some form of established process, often an agile model with iterative development using a backlog. The requirements are usually collected in a feasibility study with the customer, but it happens that the client comes with a finished requirement specification ready to be implemented. There is not much difference between an IoT project compared to an average project, it is more depending on how the customer wants to work
and the customers’ knowledge in the area. Customer involvement usually include regular meetings on a weekly basis or monthly basis, as well as regular meetings and demonstrations of each sprint and additional meetings were warranted.

Since Attentec works agile, they usually build a basic system and then build upon it with more and more functionality. Sometimes it has happened that the client wants a prototype first, which then creates. In these cases the prototype often works as a requirements’ definition as well as a prototype. The managers feeling was that problems in building IoT solutions often lie in getting a good hardware solution and integration to third party systems such as ERP or finance system, for example. From a multidisciplinary view it might be that it is the requirement specification that is the source of the difficulties. An IoT system typically have a number of things that send data over TCP or UDP, depending on the application, and then to a Web server that stores it in any kind of database, usually a NoSQL database rather than a relational database. In the data positions are common, otherwise it is typical metrics that are interesting for the application.

In the interview with manager 3 the following were mentioned. There is no difference in the work process for IoT projects and general projects. The collection of requirements is often done together with the customer, often using interview techniques to collect them. Most of the time Attentec is working in an agile process with an ordered backlog and sprints. The sprint length is often decided together with the customer.

Often the work is done in depth on some functionality first, this often causes the first sprint demo to be a bit sparse on functionality. The customer involvement varies from customer to customer, but it is often enough with the sprint meetings. New customers could have some problems with working agile. Technical interfaces and especially physical interfaces often presents some issues and a lot of things that could go wrong. Frameworks could also be a source of errors especially when trying to modify them to do something that they were not intended to be used for. The data is sensor data that is sampled at a chosen frequency, suitable for the task. The data could for example be used for trend monitoring or it could be geodata but it is almost always application specific. The storage is usually in NoSQL databases due to not knowing from the beginning what exactly the data that should be stored looks like and should be used for so that it is easy to change late on.

Summarising all interviews has led to the following results. The work process is usually agile, the teams work in iterations, sprints, after an ordered list of tasks, backlog. There is no difference between a general project and a IoT project regarding the work process. The customer can have requirements on for example what work process should be used, iteration length or their involvement. The requirement specification is often developed together with the customer. Prototypes are sometimes used, but more often the development starts without making any prototypes. The customer involvement is varying from customer to customer from daily meetings to only sprint meetings, it is often enough with the sprint meetings. The customers thoughts must often be clarified and broken down to understand what they want. This is usually a part of the development of the requirement specification.

The data from IoT systems is often stored in NoSQL databases. Positional-data or geodata has been mentioned as a common trait in IoT data in a number of interviews, often with positions together with sensor data. Trends in the data and large amounts of data were also mentioned. The hardest part of an IoT system seems to vary depending on who you ask, but a good hardware solution and integration between different systems are two frequently mentioned possible problem areas, none of which is relevant for this study.

The interviews regarding visualisations were not as conclusive as the interviews regarding work processes and IoT were. Both were projects where the visualisation was the main feature for the customer. In one the customer had a clear vision on what they wanted to see while in the second a graphical designer was used to develop mock-ups to show the customer and to use in the development. The type of graphs varied widely and the types used are de-
5.2. Pre-study

The collection and forming of data and understanding the data that should be visualised were the problems mentioned in these projects. These three problems are relevant to this study as the collection and forming of data is part of the evaluations in this study, and the understanding of data is needed to be able to create visualisations from that data.

5.2.2 Technical evaluation

Since there are no import scripts for data it is possible to import any data as long as the language supports it. Zeppelin has support for multiple languages but the evaluation was performed in Python 2.7 since the evaluator had the most experience in Python and the language choice should not affect the result. There were no problems with importing any data, the only issue was how to flatten the data to fit a SQL table style.

Zeppelin has a modifiable memory limit and a modifiable maximum number of results that can be displayed which sets a limit on the amount of data that is possible to visualise on a given machine.

Zeppelin comes with the following charts

- Scatter plot
- Column chart/Histogram
- Line charts
- Pie charts

The following charts can also be installed using the plug-in library Helium as of the 28 of March 2017.

- Heat map
- 2 x Bubble chart
- Scatterplot
- Spline
- Column range

The interaction with the built-in graphs are limited to drag and drop with columns to use the different columns on axes and values and filtering using dynamic forms in the SQL command.

Summarising the evaluation shows that there are numerous graphs that are not implemented in Zeppelin but commonly used by Attentec. No geographically based charts and maps are implemented which is one of the aspects that were mentioned in a lot of the interviews held and described in the previous section. Time series can not be visualised satisfactory either if they are not uniformly spaced since the spacing on the x-axis is fixed in the charts and not depending on the timestamp on the built in charts. The limits in data size could also be a problem with large data sets depending on the hardware on the machine running the tool. For the purpose of prototyping data visualisations the limits in data size will likely not be a problem for the computers used by the consultants at Attentec.
5.3 Implementation iteration 1

From the pre-study NoSQL and data with geographical positions were stated to be common in IoT development. In Zeppelin there is no support for visualisation of geographical data on a map if not using a scatter plot for plotting data using the x and y axes as latitude, longitude counts. In this implementation phase a visualisation plug-in was developed where this functionality was added.

The plug-in was developed using the provided guide [87] for developing a visualisation for Zeppelin. The plug-in was developed in JavaScript and used the Google Maps API to plot the data on a map. Three fields were added, North, East and Scale. North and East held the SWEREF99TM coordinates while the scale field contained the data to be visualised at the position. Options to chose to visualise the average of the points with the same coordinates, the summed value, the minimum value, the maximum value or count for the number of measurement on a given position where also added to the plug-in.

The result of the first implementation iteration was the plug-in that can be seen in figure 5.1.

5.4 Evaluation iteration 1

In the following section the results from the first evaluation phase will be presented, starting with the task creation followed by the SUS and SUM evaluations and lastly the interviews.
5.4. Evaluation iteration 1

Figure 5.2: Data from iteration 1.

5.4.1 Task creation

From the pre-study NoSQL and data with geographical positions were stated to be common in IoT development. This was therefore the basis when choosing the data and when creating the tasks.

The data chosen was weather station data collected from the Swedish Transport Administrations (Swedish: Trafikverket) API. An example of how the data looked is shown in figure 5.2. The data was collected by developing a Python script that was executed using a cron job every 30 minutes for seven days between 2016-03-28 00:00:00 and 2016-04-04 00:00:00. The data collected during the seven days was then inserted into a MongoDB collection to be used in the task.

To create fair tasks that resemble real IoT projects the data was first imported and inserted into the SQLContext in Zeppelin. The knowledge gain in the interviews about common characteristics were then used to determine the structure of the visualisation tasks. Since using the measurements together with the location data was one of the results from the pre-study, different visualisations were created and problems formulated using the measurement data.
ments and the location.

The tasks to perform with Zeppelin were the following:

1. Read the data from the local MongoDB database with the collection called weatherData format and map it to a SQLContext called trafikverket_weather_data.
   
   - Id, Name, Measurement Time, Northern and Eastern Coordinates, and Road Number should always be included in the trafikverket_weather_data, but it is not certain that they are in WeatherData. If the value is not present, the record should not be stored.
   
   - All values containing Icon can be ignored. Other values should be included if stored.
   
   - Store northern and eastern coordinates separately from the SWEREF99TM point.
   
   - Store a rounded measurement time to the next half hour.

2. Retrieve all measurement values from a road number over a period of time and display this data in the form of a table. The road number and between which dates and times the data will be retrieved should be possible to change with text fields. Use route 18 during 2017-03-29 as the default value.

3. Display the rainfall from all measurement stations between 2017-03-30 12:00:00 and 2017-03-30 12:31:00 on a map.

4. Download the road temperature from all measurement stations from Route 22 between 2017-03-28 00:00:00 and 2017-04-04 00:00:00 and show as line chart with one line per measurement station.

5. Display the precipitation type and precipitation from all measurement stations between 2017-03-30 12:00:00 and 2017-03-30 12:31:00 as points in a coordinate system based on the geographical coordinates.

6. Visualise the rainfall for the 20 rain stations with the highest temperature for one day, 2017-03-30, with a pie chart.

7. Show the average rainfall along road number 4 on a map.

8. Visualise the number of measurement stations by road number that reported measured values during the last 24 hours as a histogram.

The hypothesis for the first iteration was that Zeppelin as it is in version 0.8.0-SNAPSHOT with the default visualisations as well as with a developed map plug-in is usable from a developer’s perspective about tools perceived usability and that it would assist in the development of a visualisation of data.

5.4.2 SUS evaluation

The results from the SUS evaluation have been compiled in table 5.2. Participant 1 and participant 2 are the two master thesis students and participant 3-5 are consultants at Attentec. The average SUS value was 70 which, according to Bangor et al. as described in section 2.6.2, is borderline to acceptable. Notable is that participant 2 gave a significantly lower value than the other participants. It is also clear from the SUS evaluation that the consultant, participant 3-5, consistently found the tool more usable than the master thesis students, participant 1 and 2. If the questions are analysed one by one question 9, 3 and 5 got the lowest scores. This indicates that these areas could be improved. Those questions were:
5.4. Evaluation iteration 1

- Question 9 “I felt very confident using the system.” translated to “Jag kände mig väldigt säker i att använda systemet.”.
- Question 3 “I thought the system was easy to use.” translated to “Jag tycker att systemet var lätt att använda.”.
- Question 5 “I found the various functions in this system well integrated.” translated to “Jag finner att de olika funktionerna i system var väl integrerade.”.

<table>
<thead>
<tr>
<th>Question</th>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Participant 3</th>
<th>Participant 4</th>
<th>Participant 5</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>3.0 (4.0)</td>
</tr>
<tr>
<td>2</td>
<td>2 (3)</td>
<td>4 (1)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>3.4 (1.6)</td>
</tr>
<tr>
<td>3</td>
<td>2 (3)</td>
<td>0 (1)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>2.4 (3.4)</td>
</tr>
<tr>
<td>4</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>1 (4)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>3.2 (1.8)</td>
</tr>
<tr>
<td>5</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>2 (3)</td>
<td>4 (5)</td>
<td>2 (3)</td>
<td>2.6 (3.6)</td>
</tr>
<tr>
<td>6</td>
<td>3 (2)</td>
<td>0 (5)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>7</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>4 (5)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>3.2 (4.2)</td>
</tr>
<tr>
<td>8</td>
<td>4 (1)</td>
<td>0 (5)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>9</td>
<td>1 (2)</td>
<td>2 (3)</td>
<td>1 (2)</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>1.8 (2.8)</td>
</tr>
<tr>
<td>10</td>
<td>2 (3)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>1 (4)</td>
<td>4 (1)</td>
<td>2.8 (2.2)</td>
</tr>
</tbody>
</table>

SUS 65 47.5 67.5 87.5 82.5 70

The value without parenthesis for the odd questions in the columns is the chosen value - 1. The value without parenthesis for the even questions in the columns is the 5 - chosen value. The value in the parentheses is the actual chosen value on each question.

Table 5.2: SUS evaluation answers.

5.4.3 SUM evaluation

The results from the SUM evaluation are presented in table 5.4. The three first columns containing hardness, satisfaction and time are the respondents rating in the survey that they received after each task. The column completed contains an X if the respondent managed to solve the task in the allotted time which was tracked by the test leader. The last column, measured time, is the time that the respondent spent trying to solve the task and was also tracked by the test leader.

The SUM value were calculated as described in the method, section 4.3.5 and in the theory, section 2.6.2. The summed usability metric was calculated without the errors. 4.0 was used as the specification limit for satisfaction as recommended by Sauro and Kindlund [49]. The specification limit for the time for each task was set by the author doing the tasks after having used the tool and gaining experience in the tool. The time it took with experience of the tool multiplied by 1.5 was set as the specification limit for each task. The goal times for each task is presented in table 5.3.

The results from the evaluation were used in the calculations described in 4.3.5 and the results from the calculations are presented in table 5.5. Worth noting is the extremely low value for task 1 that was the task to import the data. That task had a caped time to 30 minutes and no participant was able to complete the task within the time span leading to 0% completion and 0% time, the task also received a low satisfaction score.

In table 5.5 the satisfaction column indicates the probability that the user would rate the satisfaction a 4 or higher on the task. The completion column indicates the percentage of the participants that completed the task of the participant that attempted to do the task. The time column indicates the probability that the user would complete the task on or under the
5.4. Evaluation iteration 1

<table>
<thead>
<tr>
<th>Task</th>
<th>Goal Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30:00</td>
</tr>
<tr>
<td>2</td>
<td>2:45</td>
</tr>
<tr>
<td>3</td>
<td>2:00</td>
</tr>
<tr>
<td>4</td>
<td>4:15</td>
</tr>
<tr>
<td>5</td>
<td>1:30</td>
</tr>
<tr>
<td>6</td>
<td>4:30</td>
</tr>
<tr>
<td>7</td>
<td>1:30</td>
</tr>
<tr>
<td>8</td>
<td>2:30</td>
</tr>
</tbody>
</table>

Table 5.3: Goal time for each task in iteration 1.

goal time specified for the task. Goal times for each task can be seen in table 5.3. The SUM column is the average of the satisfaction, completion and time column and gives an overall value for the whole task. For all columns the higher the value the better it is. Since there are no guidelines on what a good value for SUM is, it is not possible to say anything about how usable the tool is. It is however possible to compare the different values to see where the usability is perceived as the highest and lowest. In the data presented in table 5.5 task 1 got the lowest perceived usability and task 6 got the highest. The data in table 5.5 will be compared to the data from the second iteration in section 5.7.

5.4.4 Interviews

After the tasks were done or the time was up each respondent was interviewed. In this section the results from these interviews will be presented. Each respondents answers from the interview have been compiled together with things mentioned during the evaluation. The pronoun “he” will be used regardless of the gender of the respondent to preserve the respondent’s integrity.

The first participant did not succeed with the first task. He managed to create a solution for 6 of the 17 fields, among those were the splitting of the string containing the coordinates but not the rounding to the nearest half hour. For the visualisation tasks he managed to finish four out of seven before the time was up.

In the interview with participant 1 the following things were mentioned. The participant had some knowledge in Python and no prior usage of similar tools or Zeppelin. He had previously made visualisations using d3. The participant did not think that the knowledge of Python had a large impact in completing the first task. The hardest part according to the participant was creating the data mapping and importing the data. The easiest and best liked part was selecting how to visualise the data and the ease of inspecting the data and changing what to visualise. The worst part was that the tool sometimes felt slow. Zeppelin did not feel like a slower alternative compared to previous knowledge of data visualisations and Zeppelin felt like a useful tool for inspecting and prototyping according to the participant. He also stated that he would use it again if presented with a use case for the tool.

Suggested improvements:

- A “Visualise now”-button instead of trying to visualise the data directly.
- The importing of data could probably be made easier with a form or something.
- Programmatically create the graphs instead.

The second participant did not succeed with the first task. He did create the database schema with the 17 fields and he split of the string containing the coordinates but did not have time to
### Table 5.4: Evaluation results.

<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Task</th>
<th>Hardness</th>
<th>Satisfaction</th>
<th>Time</th>
<th>Completed</th>
<th>Measured Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>X</td>
<td>9:09</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14:17</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>X</td>
<td>3:28</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td>0:45</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>X</td>
<td>3:21</td>
</tr>
<tr>
<td>Participant 2</td>
<td>Task</td>
<td>Hardness</td>
<td>Satisfaction</td>
<td>Time</td>
<td>Completed</td>
<td>Measured Time</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td>10:32</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>X</td>
<td>5:25</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td>8:54</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>X</td>
<td>4:30</td>
</tr>
<tr>
<td>Participant 3</td>
<td>Task</td>
<td>Hardness</td>
<td>Satisfaction</td>
<td>Time</td>
<td>Completed</td>
<td>Measured Time</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td>6:47</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>6:27</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>3:21</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>1:38</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>5:40</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td>4:46</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td>5:32</td>
</tr>
<tr>
<td>Participant 4</td>
<td>Task</td>
<td>Hardness</td>
<td>Satisfaction</td>
<td>Time</td>
<td>Completed</td>
<td>Measured Time</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>X</td>
<td>9:00</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>2:51</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>X</td>
<td>5:57</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>X</td>
<td>3:52</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7:18</td>
</tr>
<tr>
<td>Participant 5</td>
<td>Task</td>
<td>Hardness</td>
<td>Satisfaction</td>
<td>Time</td>
<td>Completed</td>
<td>Measured Time</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>3:45</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>3:34</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td>7:03</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>4:00</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>2:27</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td>2:16</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
<td>6:13</td>
</tr>
</tbody>
</table>
5.4. Evaluation iteration 1

<table>
<thead>
<tr>
<th>Task</th>
<th>Satisfaction</th>
<th>Completion</th>
<th>Time</th>
<th>SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>4.58%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.53%</td>
</tr>
<tr>
<td>Task 2</td>
<td>44.92%</td>
<td>100.00%</td>
<td>2.73%</td>
<td><strong>49.22%</strong></td>
</tr>
<tr>
<td>Task 3</td>
<td>62.09%</td>
<td>80.00%</td>
<td>6.05%</td>
<td><strong>49.38%</strong></td>
</tr>
<tr>
<td>Task 4</td>
<td>38.66%</td>
<td>100.00%</td>
<td>26.41%</td>
<td><strong>55.02%</strong></td>
</tr>
<tr>
<td>Task 5</td>
<td>69.15%</td>
<td>100.00%</td>
<td>19.01%</td>
<td><strong>62.72%</strong></td>
</tr>
<tr>
<td>Task 6</td>
<td>57.10%</td>
<td>75.00%</td>
<td>65.97%</td>
<td><strong>66.03%</strong></td>
</tr>
<tr>
<td>Task 7</td>
<td>55.48%</td>
<td>100.00%</td>
<td>12.70%</td>
<td><strong>56.06%</strong></td>
</tr>
<tr>
<td>Task 8</td>
<td>37.75%</td>
<td>50.00%</td>
<td>0.00%</td>
<td><strong>28.58%</strong></td>
</tr>
</tbody>
</table>

Table 5.5: Calculated SUM.

add the fields to the database or do the rounding to the nearest half hour. Of the visualisation tasks he managed to complete four out of seven before his time was up.

In the interview with participant 2 the following were mentioned. The participant had good knowledge in Python and no previous experience in Zeppelin or similar tools. He did think that his knowledge in Python affected the possibility to perform the data import. The easiest and best liked part of the system according to the participant was the visualisation. Flaws in the usability was the worst part and also mentioned as the hardest part. The parts regarding text input were taking longer time than it should since the input is not a text field so the user can not right click to copy and plug-ins depending on that will not work according to the participant. The participant thought that Zeppelin is mostly a tool for data analysis and he would not have used it as a tool to prototype data visualisation, but stated that he might use it again as he liked the visualisations.

Suggested improvements:

- Make the drag and drop feature work consistently on both the field and the windows
- Box-plot diagram

The third participant did not succeed with the first task. He managed a solution for 6 of the 17 fields, among those were the splitting of the string containing the coordinates but not the rounding to the nearest half hour, he did however start to work on that problem. Of the seven visualisation tasks he managed to solve all before the time was up.

In the interview with participant 3 the following were mentioned. He had not used Zeppelin or any similar tool before, but he had done some data visualisations using gnuplot, graphviz and matplotlib earlier. The participant was comfortable in using Python and did not think that it affected the first task in any major way. The easiest part according to the participant was to run the different parts and that it was nice to be able to create the database in Python and then query it in the box below. The hardest part was to know what you could do, but when you know how to do it, it is easy. The best part was the visual appeal of the line charts and the worst part is that it is a webpage that is slow. The participant also stated that he probably would use Zeppelin again to examine data.

Suggested improvements:

- Create queries through drag and drop.
- Block Ctrl+s so that the save screen does not pop up when the user by reflex press Ctrl+s.
- Convert MongoDB to SQL in a simpler way.
- Better code editor.
The fourth participant did not succeed with the first task. He created the database schema with the 17 fields and he split the string containing the coordinates but did not have time to add the fields to the database or do the rounding to the nearest half hour. Of the seven visualisation tasks he managed to solve four.

In the interview with participant 4 the following were mentioned. The participant had used Python Notebooks before but nothing more similar to Zeppelin. He also had experience with programming in Python. He thought that the knowledge of Python did not affect the first task since the user can get the general structure from the examples. The easiest part was creating the mapping and importing of data if there had been more time to do it according to the participant. The hard part was creating more complex SQL queries. The best part was combining SQL with drag and drop of the columns to visualise. The worst part was the requirement of using SQL, it could perhaps have been done simpler. The participant did not feel that anything took longer time than expected to complete in Zeppelin and would use Zeppelin again.

Suggested improvements:
- Simplify the creation of queries.
- Simplify the mapping and importing of data.
- More visualisations.
- Adding possibility to change colour, size and shape on the map visualisation.

The fifth participant did not succeed with the first task. He created the database schema with the 17 fields and he split of the string containing the coordinates. He also managed to add 6 of the 17 the fields to the database but did not do the rounding to the nearest half hour. From the seven visualisation tasks he solved six before the time was up.

In the interview with participant 5 the following were mentioned. The participant had not used any similar tool before. He had experience in Python and thought that it affected the possibility to complete the first task somewhat. He also thought that the experience in Python makes it take less time to understand the examples. The easiest part was plotting the data on maps and the visualisations and the hardest part was the mapping and importing of data according to the participant. The ease of plotting and visualising the data were mentioned as the best part and the error messages as the worst part of Zeppelin. The participant also stated that he thought that the tool was good for creating prototypes, that he thought that Zeppelin was a useful tool and that he would use it again.

Suggested improvements:
- Show line numbers.
- Improve the error messages so that they display the correct line number.
- Add more aspects to the map visualisation such as colour and size.

In summary the participants were mostly satisfied with the visualisations and the interface for creating the visualisations. Most of the participants said that their knowledge in Python had none or a small impact in the first task. None of the participants completed the first task on time. All participants did approximately half the first task.

In three of the interviews the import of data was mentioned as something that could probably be made easier. Two participants mentioned adding features to the map plug-in and two participants mentioned simplifying the creation of SQL queries for example by drag and drop. A number of things were also mentioned about the code editor:
5.5 Implementation iteration 2

Mapping and importing the data from a MongoDB database were chosen as the first feature to improve. This was done for two reasons. The first was that the evaluation in iteration 1 had a very low SUM score, found in section 5.4.3, on the task of mapping and importing data. It was also mentioned three times in the interviews, found in section 5.4.4, that the mapping and import of data should be made simpler.

The map plug-in developed in the first implementation found in section 5.3 was the second feature to be improved. This was done by adding the possibility to change colour and add grouping on a field. The reason for the improvements of the map visualisation was that it was mentioned twice in the interviews, found in section 5.4.4.

Some of the code editor improvements suggested did already have existing solutions. Two of them were the showing of line numbers and code completion by hitting “Ctrl + .”. An editor improvement that did not exist was to block “Ctrl + s”. Therefore, the creation of a script that blocks “Ctrl + s” so that the save screen does not pop up when the user by reflex press “Ctrl + s” was the third priority in this phase.

To simplify the creation of SQL queries was deemed to be too complex to fit in the scope of the second implementation phase. It was therefore not improved.

5.5.1 Import from MongoDB

Since the participants in the interviews that suggested creating a simplified way of importing data also showed concerns about losing the possibility to modify the data by simplifying the import, the choice of creating a script that generates the Python code for importing data was made. That way the importing of data from a MongoDB collection will be simplified without removing the possibility for modifications of the data in the import script. Therefore, a Python script was developed that used the built-in forms in Zeppelin for choosing database, credentials, collection and filter. The script then iterates through a user defined number of records to build a list of all stored fields, a database schema. The schema is then presented in a table and the user can chose which fields that should be included and the data type of the field as well as if they should be nullable or have a default value. Using JavaScript a preview of the generated code is shown and a button that uses the Zeppelin API to generate a new note with the code and remove the form if pressed. A screenshot of the import script can be seen in figure 5.3.

5.5.2 Improving the geospatial visualisation plug-in

The map plug-in developed in the first implementation found in section 5.3 was extended to add the possibility to change colour and add grouping on a field. The improvement of the geospatial visualisation was made using JavaScript. The plug-in was improved to store the aggregation between page loads and to follow the same layout as the rest of the chart plug-ins. Fields for storing parameters regarding the graph were also added with the option to store the parameters between sessions. The parameters added were:

- Coordinate system (Options: SWEREF99TM, Decimal Degrees).
5.5. Implementation iteration 2

Figure 5.3: Image of the code generation plug-in developed during implementation 2.

- Map type (Options: Terrain, Satellite, Roadmap, Hybrid).
- Fill opacity of the map markers.
- Scale of the markers.
- Colouring of the markers.

Adding the option to choose between the two coordinate systems were done since decimal degrees is a common coordinate system and supporting both improves the number of use cases for which the plug-in can be used. Colouring and size of the markers were added to support scales with other colours than green, yellow and red which the first implementation supported. The option of changing colours also improves the usability since colour-blind persons can modify which colours the visualisation should use. A screenshot of the improved map plug-in can be seen in figure 5.4.

To add the possibility of grouping on a column and scale the size of the markers an option between Scale and Group and scale were added. The Group and scale option has a fourth field for grouping. The grouping is used for colouring the markers and the scale field is used to determine the size of the markers. A screenshot of the Group and scale chart can be seen in figure 5.5.
5.5.3 Improving code editor

A JavaScript blocking the default behaviour of “Ctrl + s” was written and added to Zeppelin. This was done since the likelihood of the user pressing “Ctrl + s” and wanting to save an HTML copy of Zeppelin was deemed very small. If the user wanted to save an HTML copy of Zeppelin, it is still possible to do by using the menu in the web-browser.

5.6 Evaluation iteration 2

In the following section the results from this phase will be presented, starting with the task creation followed by the SUS and SUM evaluations and lastly the interviews.
5.6. Evaluation iteration 2

5.6.1 Task creation

When creating this task data was joined from two sources, train stations data and train announcements to create a similar data structure as in the data importing task in evaluation 1. The reason for using new data was to minimise the risk that the respondents remembered how they solved the task last time. With new data they were not helped even if they remembered what fields they used last time.

The data collected was train announcements and station data collected from the Swedish Transport Administrations (Swedish: Trafikverket) API. The data was collected by modifying the Python script developed in iteration 1 and it was then executed using a cron job every 30 minutes for 2 days between 2017-04-07 00:00:00 and 2017-04-09 00:00:00. Afterwards the train announcements and station data were joined together to give the data structure illustrated in figure 5.6. The data collected was then inserted into a MongoDB collection to be used in the tasks. The same number of fields were used for the first task in both iterations so that it would not affect the time to complete the task.

In the first iteration the participant had to round a time to the nearest half hour, that was not included in the second iteration but the participant should instead calculate the difference between two timestamps and store the difference in minutes. Both tasks were deemed to have the same level of difficulty.

The visualisation tasks, 2-8, were based on the tasks in iteration 1 to have the same moments and complexity. One change in the tasks was that question 5 was changed to use the newly developed part of the map plug-in instead of a scatter plot while keeping the complexity of the task.

The task to perform with Zeppelin in evaluation 2 were the following:

1. Read the data from the local MongoDB database traffic network with the collection called trainData format and map it to an SQLContext called trainData.
   - The following fields shall be included in the record: ActivityId, ActivityType, AdvertisedTimeAtLocation, AdvertisedTrainIdent, Cancelled, EndStation.AdvertisedLocationName, InformationOwner, Location.AdvertisedLocationName, Location.Geometry.SWEREF99TM, OtherInformation, ProductInformation, ScheduledDepartureDateTime, StartStation.AdvertisedLocationName, TimeAtLocation, TypeOfTraffic.
   - If data for one of the fields is not available, it can be set to null.
   - SWEREF99TM positions should be divided into a northern and an eastern part.
   - A time difference between TimeAtLocation and AdvertisedTimeAtLocation shall be stored in minutes. If no data from TimeAtLocation or AdvertisedTimeAtLocation exist, the database row shall not be saved.

2. Retrieve all data from a station over a period of time, based on the advertised time of the location, and display this data in the form of a table. The station and between which dates and times the data should be retrieved from should be possible to change with text fields. Use station Linköping C between 2017-04-07 04:00:00 and 2017-04-07 08:00:00 as default value.

3. Show the average delay at all stations for all trains between 2017-04-07 06:00:00 and 2017-04-07 07:00:00 on a map.

4. Display delays from Stockholm Central between 2017-04-07 00:00:00 and 2017-04-09 00:00:00 as a line diagram.
Figure 5.6: Example data iteration 2.
5. Show the delay and end station from all stations between 2017-04-07 07:00:00 and 2017-04-07 08:00:00 as points in a coordinate system based on the geographic coordinates on a map. The colour should indicate the final station and the size, the length of the delay.

6. Visualize the average delay of the 20 stations with the highest average delay between 2017-04-07 15:00:00 and 2017-04-07 19:00:00, with a pie chart. Use GROUP BY in the SQL query.

7. Show average delay for train number 8713 on a map.

8. Show average delay from Stockholm Central per company, information owner, as histogram.

The hypothesis for the second iteration was that Zeppelin as it is in version 0.8.0-SNAPSHOT with the default visualisations as well as with an improved map plug-in and a code generation script for importing data has improved the usability since the previous iteration from a developer’s perspective about tools perceived usability.

5.6.2 SUS evaluation

The participants are the same as in iteration 1 and have the same number so that the changes can be compared on an individual level. Participant 1 and participant 2 are the two master thesis students and participant 3-5 are consultants at Attentec.

The results from the SUS evaluation can be seen compiled in table 5.6. The average SUS score value was 68.5 which, according to Bangor et al. as described in section 2.6.2, is borderline to acceptable. Notable is that participant 4 gave a lower value in this evaluation than in the previous and still managed to complete more tasks and was generally more positive during the interview.

When looking at the different questions one by one, question 1 got the lowest scores. That question was “I think that I would like to use this system frequently.” translated to “Jag tror att jag skulle vilja använda systemet frekvent.”.

<table>
<thead>
<tr>
<th>Question</th>
<th>Person 1</th>
<th>Person 2</th>
<th>Person 3</th>
<th>Person 4</th>
<th>Person 5</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 (2)</td>
<td>1 (2)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>2.4 (3.4)</td>
</tr>
<tr>
<td>2</td>
<td>3 (2)</td>
<td>2 (3)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>3</td>
<td>3 (4)</td>
<td>1 (2)</td>
<td>4 (3)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>2.8 (3.8)</td>
</tr>
<tr>
<td>4</td>
<td>2 (3)</td>
<td>4 (1)</td>
<td>1 (4)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>3.0 (2.0)</td>
</tr>
<tr>
<td>5</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>2.8 (3.8)</td>
</tr>
<tr>
<td>6</td>
<td>3 (2)</td>
<td>1 (4)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2.6 (2.4)</td>
</tr>
<tr>
<td>7</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>3 (4)</td>
<td>4 (3)</td>
<td>4 (5)</td>
<td>3.2 (4.2)</td>
</tr>
<tr>
<td>8</td>
<td>3 (2)</td>
<td>1 (4)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>2.6 (2.4)</td>
</tr>
<tr>
<td>9</td>
<td>2 (3)</td>
<td>2 (3)</td>
<td>3 (3)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>2.6 (3.6)</td>
</tr>
<tr>
<td>10</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2 (3)</td>
<td>3 (2)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>SUS</td>
<td>65</td>
<td>50</td>
<td>65</td>
<td>77.5</td>
<td>85</td>
<td>68.5</td>
</tr>
</tbody>
</table>

The value without parenthesis for the odd questions in the columns is the chosen value - 1.
The value without parenthesis for the even questions in the columns is the 5 - chosen value.
The value in the parentheses is the actual chosen value on each question.

Table 5.6: SUS evaluation answers.

5.6.3 SUM evaluation

The results from the evaluations are presented in table 5.8. The results from the evaluation were used in the calculations described in 4.3.7 and the results from the calculations are pre-
sent in table 5.9. 4.0 was used as the specification limit for satisfaction as recommended by Sauro and Kindlund [49]. The specification limit for the time for each task was set by the author doing the tasks after having used the tool and gaining experience in the tool. The time it took with experience of the tool multiplied by 1.5 was set as the specification limit for each task. The goal times for each task is presented in table 5.7.

<table>
<thead>
<tr>
<th>Task</th>
<th>Goal Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24:00</td>
</tr>
<tr>
<td>2</td>
<td>5:15</td>
</tr>
<tr>
<td>3</td>
<td>3:00</td>
</tr>
<tr>
<td>4</td>
<td>3:45</td>
</tr>
<tr>
<td>5</td>
<td>4:15</td>
</tr>
<tr>
<td>6</td>
<td>5:09</td>
</tr>
<tr>
<td>7</td>
<td>2:30</td>
</tr>
<tr>
<td>8</td>
<td>3:00</td>
</tr>
</tbody>
</table>

Table 5.7: Goal time for each task in iteration 2.

Question 4 was unfortunately made a bit simpler than in iteration 1. This was due to a human error that missed adding information about grouping on the activity type in the question. The result for this question can therefore be a bit better than what would be expected.

In table 5.9 the satisfaction column indicates the probability that the user would rate the satisfaction a 4 or higher on the task. The completion column indicates the percentage of the participants that completed the task of the participant that attempted to do the task. The time column indicates the probability that the user would complete the task on or under the goal time specified for the task. Goal times for each task can be seen in table 5.7. The SUM column is the average of the satisfaction, completion and time column and gives an overall value for the whole task. Since there are no guidelines on what a good value for SUM is, it is not possible to say anything about how usable the tool is. For all columns the higher the value the better it is. It is however possible to compare the different values to see where the usability is perceived as the highest. It is also possible to compare the results with the results in iteration 1, a comparison is presented in section 5.7.2.

5.6.4 Interviews

After the tasks were done or the time was up each respondent was interviewed. In this section the results from these interviews will be presented. Each respondents answers from the interview have been compiled together with things mentioned during the evaluation. The pronoun “he” will be used regardless of the gender of the respondent to preserve the respondent’s integrity.

The first participant spent the first 8 minutes choosing the correct values and if they should be nullable. He then spent the next 12 minutes splitting the coordinates and calculating the time difference. He did however use the wrong path for the coordinates resulting in no entries being stored in the database and with 2 minutes left he found the error but had a second error in the time difference that he did not fix within the given 30 minutes. With a few extra minutes he would most likely have succeeded with the first task.

In the interview with participant 1 the following were mentioned. He thought that it went better than the last evaluation. The easiest parts were the visualisations and the hardest part was still the importing of data even though he thought that it was easier than in the first evaluation. The best part of the tool was that it was easy and quick to visualise data and find interesting visualisations. The worst part and something that could be further improved was the importation of data. The participant stated that he missed a feature of displaying database records from the MongoDB such as the example data presented in the task.
<table>
<thead>
<tr>
<th>Participant 1</th>
<th>Task</th>
<th>Hardness</th>
<th>Satisfaction</th>
<th>Time</th>
<th>Completed</th>
<th>Measured Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>X</td>
<td></td>
<td>4:23</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>4:23</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>X</td>
<td></td>
<td>5:10</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>3:05</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4:00</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>1:48</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:21</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Participant 2</th>
<th>Task</th>
<th>Hardness</th>
<th>Satisfaction</th>
<th>Time</th>
<th>Completed</th>
<th>Measured Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td></td>
<td></td>
<td>30:00</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td></td>
<td>10:35</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td></td>
<td>4:54</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td></td>
<td>2:21</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td></td>
<td>3:45</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td></td>
<td>8:40</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Participant 3</th>
<th>Task</th>
<th>Hardness</th>
<th>Satisfaction</th>
<th>Time</th>
<th>Completed</th>
<th>Measured Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>X</td>
<td></td>
<td>28:46</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td></td>
<td>6:40</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:15</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>3:56</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:10</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>X</td>
<td></td>
<td>4:46</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>1:36</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>1:40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Participant 4</th>
<th>Task</th>
<th>Hardness</th>
<th>Satisfaction</th>
<th>Time</th>
<th>Completed</th>
<th>Measured Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td></td>
<td>29:45</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>X</td>
<td></td>
<td>10:15</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>X</td>
<td></td>
<td>8:10</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:55</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>X</td>
<td></td>
<td>2:40</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>X</td>
<td></td>
<td>7:25</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Participant 5</th>
<th>Task</th>
<th>Hardness</th>
<th>Satisfaction</th>
<th>Time</th>
<th>Completed</th>
<th>Measured Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>X</td>
<td></td>
<td>24:31</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>3:45</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>4:08</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:53</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>X</td>
<td></td>
<td>4:05</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:53</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:58</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>X</td>
<td></td>
<td>2:43</td>
</tr>
</tbody>
</table>

Table 5.8: Evaluation results table.
thought that Zeppelin was a good and usable tool as a developer when looking at data and interesting visualisations. The participant stated that he thought that the changes between the first and the second iteration absolutely improved the usability. He also said that he would use Zeppelin again if presented with a similar task.

The second participant clicked the convert to code button directly and the test was interrupted to open the form again for the code generation. During the second attempt he filled out the form correctly but instead of copy the code for splitting the coordinates from the example code he spent around 15 minutes to solve it by himself leaving no time to solve the time difference part within the 30 minutes. Of the seven visualisation tasks he managed to solve five.

In the interview with participant 2 the following were mentioned. He thought that the knowledge of Python still affected the possibility of doing the first task but not as much as in the previous iteration. The easiest part was using SQL to get the data and create a nice graph. The hardest part was not being able to use personalised keyboard shortcuts since the text editor is not a text field. The best parts were the graphs and that the user do not have to copy the data to Excel to create a visualisation which is good since they do not get the data limits from Excel and can work on large data sets. The worst part was the text editor. The participant stated that he would rather write the code in a text editor and make the mapping and importing of data outside of Zeppelin and instead only use the visualisation parts. He thought that the geospatial visualisation improved the usability and that the improvements to the data importation did improve the ease of use and usability but that it might be a small use case to convert the data from MongoDB to the SparkSQL database. Aside from the text editor he thought that Zeppelin was a usable tool for creating visualisations. He stated that he would never use it as a text editor but perhaps for creating data visualisations.

The third participant did succeed with the first task. He was fairly quick to select the data in the form but did a mistake with the type of data for the advertised train identity. After 10 minutes he had added the coordinates in both the schema and the code splitting the coordinates. The next 12 minutes were used to debug the code to find the faulty data type. When that was found he solved the time difference by trial and error to get the minutes completing the task with about a minute to spare.

In the interview with participant 3 the following were mentioned. The Python knowledge affected the first task less than the last time. The easiest parts were the visualisations, the map plug-in was mentioned as the most simple visualisation to use. The hardest parts were writing the SQL queries to get the data wanted. The best part was that it was possible to do some logic in the GUI instead for writing more advanced queries. The worst part was that the tool was running in a web browser and that it gets slow with many notes. The participant suggested using tabs for organising the notes and that a desktop application perhaps would do a better job. A graphical query builder for the SQL queries was also suggested as some-
thing that would be nice to have. Statistical functions was also mentioned as a function that would be nice to have. Aside from the flaws in the editor the tool was performing as expected according to the participant. He did not think that the improvements made did affect the usability notable aside from blocking “Ctrl + s” but mentioned that the importing was faster, he also mentioned that the improvements to the map added good functionality but that it did not affect the usability. The participant stated that the tool is a usable tool that would assist in visualising data and that he would use it again since it is easier than to use for example GNUplot for the task. He also stated that he only would use it as a tool for prototyping data not as an end product.

The fourth participant was quick to click on the convert to code button in the code generation script without reading through the task completely. He therefore missed a few things and had to correct some things manually. As an example did he have to change some data types that he did not check and also had to add a field that he missed. He also ignored the information about what fields that needed to be nullable which led to further problems. He did however manage to complete the task within the time span of 30 minutes. Of the seven visualisation tasks the participant completed five.

In the interview with participant 4 the following were mentioned and noted. The easiest part was creating the mapping and importing of data according to the participant. The hard parts were to create the complex SQL queries. The best part was that it was easy to set up the table and to be able to use SQL and then test different visualisations with drag and drop of the columns to visualise. The worst part was the requirement of using SQL. This could perhaps have been done simpler for the easier queries. The participant did not feel that anything took longer time than expected to complete in Zeppelin and thought that it was a useful tool. He thought that the data importing script perhaps could have been even simpler and suggested the data type. The participant stated that he thought that the improvements to the importing of data made the tool better and more usable and that the improvements on the map plug-in perhaps was a bit hard to understand. He also said that he would use Zeppelin again.

The fifth participant filled out the form and converted the form to code after 5 minutes having chosen the correct data types and nullable. After an additional 2 minutes he had solved the splitting of the coordinates, after 9 and a half minutes he had added the time difference but had not added the fields to the database schema and had some trouble with skipping the data record if no time difference could be calculated. After 20 minutes, all but converting the time difference to minutes was solved. Changing the time difference took another 4 minutes. After 24 minutes, he was finished with the task. Of the visualisation tasks he managed to complete all in the allocated time frame.

In the interview with participant 4 the following were mentioned and noted. The participant said that his knowledge in Python did affect the task some, since Python datetime is a bit tricky to work with. The map visualisations were mentioned as the easiest and the best parts. The hardest was having a person watching him when working. He stated that the worst part was the error messages that have the wrong line number. The text editor was mentioned as a part that can be improved even though that the editor was better than expected but not good enough. The participant did not feel that anything took longer time than expected to complete in Zeppelin. The participant thought that the improvements definitely made the tool more usable. The import script was mentioned as the best improvement since it reduced the otherwise repetitive work of mapping the data. He stated that Zeppelin was a useful tool for creating data visualisations and that he would use it again since there exist a large number of ready visualisations in the tool.

Three out of five succeeded with the first task and another one was very close in succeeding. The part of the first task that most participants had problems with was checking if they had chosen the correct data type. This resulted in that three out of five had at least one issue intro-
duced due to the data type. To calculate the time difference in the first task was also a source of problems and the part in which most of the participants thought that their knowledge in Python affected the outcome.

The visualisations seemed to be the easiest parts of the tool, especially the map visualisation since it was mentioned as the easiest part by two participants. On what the hardest part was there were no consensus but to create the more advanced SQL queries was mentioned twice. The best part seemed to be that the tool can do visualisations quickly. The worst part and how Zeppelin could be improved also seemed to vary but the creation of SQL queries and the text editor were mentioned the most. All participants stated that they thought that Zeppelin was a usable tool with the possible exception for the text editor in Zeppelin. All participants also stated that they would use Zeppelin again for similar visualisation tasks.

So in summary, the modifications did improve the usability for the use cases tested according to all participants even though some thought that the improvements were larger than others.

### 5.7 Compilation of the evaluations

In this section a compilation of the results from the both evaluations in iteration 1 and 2 will be presented to highlight differences and similarities.

#### 5.7.1 SUS

<table>
<thead>
<tr>
<th>Question</th>
<th>Person 1</th>
<th>Person 2</th>
<th>Person 3</th>
<th>Person 4</th>
<th>Person 5</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 iteration 1</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>3.0 (4.0)</td>
</tr>
<tr>
<td>1 iteration 2</td>
<td>1 (2)</td>
<td>1 (2)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>2.4 (3.4)</td>
</tr>
<tr>
<td>2 iteration 1</td>
<td>2 (3)</td>
<td>4 (1)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>3.4 (1.6)</td>
</tr>
<tr>
<td>2 iteration 2</td>
<td>3 (2)</td>
<td>2 (3)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>3 iteration 1</td>
<td>2 (3)</td>
<td>0 (1)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>2.4 (3.4)</td>
</tr>
<tr>
<td>3 iteration 2</td>
<td>3 (4)</td>
<td>1 (2)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>2.8 (3.8)</td>
</tr>
<tr>
<td>4 iteration 1</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>1 (4)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>3.2 (1.8)</td>
</tr>
<tr>
<td>4 iteration 2</td>
<td>2 (3)</td>
<td>4 (1)</td>
<td>1 (4)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>3.0 (2.0)</td>
</tr>
<tr>
<td>5 iteration 1</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>2 (3)</td>
<td>4 (5)</td>
<td>2 (3)</td>
<td>2.8 (3.8)</td>
</tr>
<tr>
<td>5 iteration 2</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>2.6 (3.8)</td>
</tr>
<tr>
<td>6 iteration 1</td>
<td>3 (2)</td>
<td>0 (5)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>4 (1)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>6 iteration 2</td>
<td>3 (2)</td>
<td>1 (4)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>2.6 (2.4)</td>
</tr>
<tr>
<td>7 iteration 1</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>4 (5)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>3.2 (4.2)</td>
</tr>
<tr>
<td>7 iteration 2</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>3 (4)</td>
<td>4 (5)</td>
<td>3 (4)</td>
<td>3.2 (4.2)</td>
</tr>
<tr>
<td>8 iteration 1</td>
<td>4 (1)</td>
<td>0 (5)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>8 iteration 2</td>
<td>3 (2)</td>
<td>1 (4)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2.6 (2.4)</td>
</tr>
<tr>
<td>9 iteration 1</td>
<td>1 (2)</td>
<td>2 (3)</td>
<td>1 (2)</td>
<td>3 (4)</td>
<td>2 (3)</td>
<td>1.8 (2.8)</td>
</tr>
<tr>
<td>9 iteration 2</td>
<td>2 (3)</td>
<td>2 (3)</td>
<td>3 (3)</td>
<td>3 (4)</td>
<td>3 (4)</td>
<td>2.6 (3.6)</td>
</tr>
<tr>
<td>10 iteration 1</td>
<td>2 (3)</td>
<td>3 (2)</td>
<td>4 (1)</td>
<td>1 (4)</td>
<td>4 (1)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>10 iteration 2</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>3 (2)</td>
<td>2 (3)</td>
<td>3 (2)</td>
<td>2.8 (2.2)</td>
</tr>
<tr>
<td>SUS iteration 1</td>
<td>65</td>
<td>47.5</td>
<td>67.5</td>
<td>87.5</td>
<td>82.5</td>
<td>70</td>
</tr>
<tr>
<td>SUS iteration 2</td>
<td>65</td>
<td>50</td>
<td>65</td>
<td>77.5</td>
<td>85</td>
<td>68.5</td>
</tr>
</tbody>
</table>

The value without parenthesis for the odd questions in the columns is the \textit{chosen value} - 1.
The value without parenthesis for the even questions in the columns is the 5 - chosen value.
The value in the parentheses is the actual \textit{chosen value} on each question.

Table 5.10: SUS evaluation answers iteration 1 and 2.

The SUS score did not change significantly between the first and second iteration as can be seen in table 5.10. The fourth participant changed his result the most by choosing values
closer to the middle value of 3, causing the decrease in the average SUS score comparing the two iterations. Question 1 and 2 decreased the most between the first and second evaluation while question 3 and 9 increased the most.

- Question 1 “I think that I would like to use this system frequently.” translated to “Jag tror att jag skulle vilja använda systemet frekvent.”.
- Question 2 “I found the system unnecessarily complex.” translated to “Jag tycker systemet var onödigt komplex.”.
- Question 3 “I thought the system was easy to use.” translated to “Jag tycker att systemet var lätt att använda.”.
- Question 9 “I felt very confident using the system.” translated to “Jag kände mig väldigt säker i att använda systemet.”.

The increase in question 3 and 9 suggest that the system increased the usability. The decrease on the first question could be attributed to the fact that the consultants does other things than just visualisations and as most of them stated would use the tool again for visualisations but that is not included in all assignments. The increased complexity of the system in question 2 could be caused by the increased options given with the import script and options in the map visualisation.

5.7.2 SUM

<table>
<thead>
<tr>
<th>Task 1 iteration 1</th>
<th>Satisfaction</th>
<th>Completion</th>
<th>Time</th>
<th>SUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1 iteration 2</td>
<td>4.58%</td>
<td>0.00%</td>
<td>0.00%</td>
<td>1.53%</td>
</tr>
<tr>
<td>Task 2 iteration 1</td>
<td>15.72%</td>
<td>60.00%</td>
<td>9.27%</td>
<td>28.33%</td>
</tr>
<tr>
<td>Task 2 iteration 2</td>
<td>44.92%</td>
<td>100.00%</td>
<td>2.73%</td>
<td>49.22%</td>
</tr>
<tr>
<td>Task 3 iteration 1</td>
<td>50.00%</td>
<td>100.00%</td>
<td>27.78%</td>
<td>59.26%</td>
</tr>
<tr>
<td>Task 3 iteration 2</td>
<td>62.09%</td>
<td>80.00%</td>
<td>6.05%</td>
<td>49.38%</td>
</tr>
<tr>
<td>Task 4 iteration 1</td>
<td>65.26%</td>
<td>100.00%</td>
<td>20.54%</td>
<td>61.93%</td>
</tr>
<tr>
<td>Task 4 iteration 2</td>
<td>38.66%</td>
<td>100.00%</td>
<td>26.41%</td>
<td>55.02%</td>
</tr>
<tr>
<td>Task 5 iteration 1</td>
<td>92.14%</td>
<td>100.00%</td>
<td>60.58%</td>
<td>84.24%</td>
</tr>
<tr>
<td>Task 5 iteration 2</td>
<td>69.15%</td>
<td>100.00%</td>
<td>19.01%</td>
<td>62.72%</td>
</tr>
<tr>
<td>Task 6 iteration 1</td>
<td>85.43%</td>
<td>100.00%</td>
<td>92.07%</td>
<td>92.50%</td>
</tr>
<tr>
<td>Task 6 iteration 2</td>
<td>57.10%</td>
<td>75.00%</td>
<td>65.97%</td>
<td>66.03%</td>
</tr>
<tr>
<td>Task 7 iteration 1</td>
<td>46.40%</td>
<td>80.00%</td>
<td>38.18%</td>
<td>54.86%</td>
</tr>
<tr>
<td>Task 7 iteration 2</td>
<td>55.48%</td>
<td>100.00%</td>
<td>12.70%</td>
<td>56.06%</td>
</tr>
<tr>
<td>Task 8 iteration 1</td>
<td>64.25%</td>
<td>100.00%</td>
<td>69.82%</td>
<td>78.02%</td>
</tr>
<tr>
<td>Task 8 iteration 2</td>
<td>37.75%</td>
<td>50.00%</td>
<td>0.00%</td>
<td>28.58%</td>
</tr>
<tr>
<td>Task 9 iteration 1</td>
<td>62.99%</td>
<td>100.00%</td>
<td>92.03%</td>
<td>85.01%</td>
</tr>
</tbody>
</table>

Table 5.11: Calculated SUM iteration 1 and 2.

In all tasks except for task 6 all measurements got better values as seen in table 5.11. Task 6 was perhaps the most tricky question since it required using a group by in the SQL query, this was stated in the second evaluation but not in the first but it still required the use of group by. The improvement for task 8 was partly due to that more than one participant having time to complete the task making it possible to calculate the time part of the SUM Score. Question 4 was unfortunately made a bit simpler due to missing to add information about grouping on the activity type in second iteration due to a human error. This was probably partly attributing to the large increase on both the time and the satisfaction of that task.

Disregarding task 4 and 8 the largest increases were for Task 1 and 5. Task 1 was importing and mapping the data and task 5 was showing the delay and end station on a map. Task
1 and 5 were also the tasks that were affected of the improvements made in the second implementation iteration described in section 5.5. This indicates that the changes made in the second iteration improved the usability of the system.

5.7.3 Interviews

The modification did improve the usability for the use cases tested according to all participants even though some thought that the improvements were larger than others. There are issues in Zeppelin that were mentioned in both iterations such as the text editor and that there could be an easier way to create the SQL queries. The most liked part of Zeppelin through both the iterations were the data visualisation and how easy it is to drag and drop the fields to visualise. The map visualisation was mentioned as the most liked chart in both iterations.
In this chapter the method and the results from the different parts of the study will be discussed and evaluated. The decisions that might have affected the outcome will be highlighted and motivated. A critically review of the following subjects will be presented in this chapter: prototyping using Zeppelin, the interpretations of usability, the choice of the prototyping tool, and how the evaluations were performed. The work done in this study in a wider context will then be discussed. The chapter is concluded with a section on how this study could be continued if there were more resources and time available. This section can also serve as a starting point for future work on the topic.

6.1 Prototyping

There is not always a need for prototyping visualisations and sometimes all prototyping that is needed is a simple drawing on a paper. In such cases the time to prototype the visualisations in Zeppelin or similar tool is not worth the cost. As mentioned in the motivation for this study, section 1.1, Zeppelin has been examined as “[a tool] for aiding in the development of prototypes for customers requiring data-intensive visualisation applications”. This does not mean that it should be used for every project, only where it is suitable with regards to the advantages and disadvantages. Zeppelin becomes useful as a tool for prototyping visualisations when the customer is not sure what they want or if the usefulness of a visualisation is in question.

During this study data from MongoDB, MySQL, CSV files and data from internet in formats of JSON, XML and CSV data have been tested in Zeppelin even if not all examples have been used or explained in this report. Importing and making small modification of the data in Zeppelin takes approximately between 15 and 90 minutes for the examples tested during this study. The time is of course depending on multiple things like the number of fields in the data and if any modifications needs to be done to the data. The data visualisations then takes approximately 1 to 15 minutes depending on the complexity of the visualisation. This is given that the data is accessible from the same computer that is running Zeppelin.

Zeppelin should be used if the time it takes to import the data and create the visualisations could be defended by the value it provides. Since time is the largest disadvantage in creating prototypes the advantages must outweigh the disadvantage. The main advantages with using Zeppelin to create prototypes is that it is possible to evaluate multiple design con-
cepts and that it can be used as a proof-of-concept for visualisations. The data analytics part of Zeppelin could also be useful to examine the data to find characteristics in the data that is of interest to visualise but is not obvious before having visualised the data.

6.2 Usability

In this study the perceived usability by the consultants was what the research question ask about. To measure the usability and to be able to compare between the two iterations and to find areas where the usability could be improved a SUS and a SUM evaluation were conducted and a semi-structured interview after each evaluation. There exist numerous other ways of evaluating the usability such as Purdue Usability Testing Questionnaire [88], Software Usability Measurement Inventory [89] or using Nielsens heuristics [90].

The interviews were conducted to be able to catch the participants thoughts on the tool. The interviews were semi structured which was a good way of keeping the discussion on topic yet being able to ask follow up question if needed. An alternative that perhaps could have captured even more of the participants thoughts would have been unstructured interviews with only discussion points. The risk of influencing the respondent or ask leading questions is greater in an unstructured interview but it is possible that the respondent would have answered more freely [59].

6.3 Choosing the tool

The chosen tool, Apache Zeppelin, is a fairly new software. Since it is a new tool, not much is written about it. Zeppelin is also still under development with lots of changes and improvements in development. This makes it possible that the changes made in this study will be solved in the tool in the future. One reason for only doing plug-ins and scripts as improvements is to be able to get updates to the core software and not having to maintain changes.

6.4 Evaluations

SUS and SUM were chosen since they were described as simple and yet having good reliability. However, the small sample of evaluators makes the values uncertain and the result is perhaps best seen as an indication.

The data used in the first iteration was weather data from multiple weather stations in Sweden. Since the different weather station had different measurement times and different gear this resulted in data that did not look the same from all stations. This weather data is representative for IoT and are used as an example in multiple studies [3,15,16,17].

In the second iteration train announcements were used to calculate and visualise delays. This data also has some of the characteristics of IoT data but is not as common as an example. In a smart city or for smart transportation it would be important to be able to predict delays and visualise delays in a similar way making the data in the second iteration a possible IoT use case.

The tasks given were based on the information from the interviews about requirements and visualisations in the pre-study and the data for each evaluation to create realistic tasks to perform. There is a risk that the tasks given during the evaluation does not correspond to the requirements that would have been given by a customer. The tasks given during the evaluation were very specific down to the type of graph to be used. This is likely not the detail level that the customer would specify the requirements in, unless the customer already knows what they want. This simplification of specifying each task down to the chart type was done to be able to compare the results between each participant and to be able to define when the task was completed. The usability should not be affected by if the tasks were presented as if a customer would have stated them or if presented like tasks during the evaluations.
The questions asked in the interviews that were not directly connected to the usability, like if something was missing or if some part could be improved, could have been affected by having other tasks that for example were less specified. The risk of that was deemed small since choosing what chart to use and possibly having to change to another chart is a trivial task in Zeppelin and not likely to affect the result.

The time given on the first task to import the data was perhaps a bit too short. The short time made it so that no respondent completed the first task in the first evaluation and only three out of five in the second. The time constraint could therefore have had a negative influence on the user's perception of Zeppelin. This in turn can have led to lower ratings of the usability due to disappointment of not finishing the task on time.

The time constraint was in place for two reasons, limited time with the consultants and that importing data for usage in a prototyping tool must be fast. The need to import data fast is necessary for the developer and customer to feel that the usage of the tool is not a waste of time and that the time spent prototyping visualizations is well spent.

In the second evaluation the fourth participant gave a lower value in the SUS evaluation than in the first evaluation and still managed to complete more tasks and was generally more positive during the interview. The participant seemed preoccupied during the second iteration and seemed to have other things in thought which could have affected the evaluation. Another reason could have been that the reference scale used by the participant to score the usability was different than in the first evaluation. A reference scale in this context is that he had lower expectations and requirements in the first iteration than in the second.

### 6.5 Work in a wider context

In this section the impact of this study in a wider context with social, environmental and ethical aspects will be discussed. Starting with how the usage of Zeppelin will affect the consultants and customers followed by environmental aspects and lastly a brief discussion about the ethical aspects of this study.

For the consultants and customers the work process will be changed for projects where the use of Zeppelin could improve the normal work process. Instead of starting to implement the visualizations directly or draw prototypes on paper or in an image processor a meeting between the customer and the consultants will take place where the prototypes are created and discussed. This meeting will likely contribute to more meeting time in the beginning of the project than in the normal work process. This should in turn provide clearer requirements that will lead to fewer meetings and changes at the end of the project.

There are some environmental aspects of using Zeppelin compared to prototyping on paper. A study of the carbon footprint of paper concluded that one A4 sheet correspond to between 4.29 and 4.74g carbon dioxide per A4 sheet [91]. Swedish energy (Swedish: Svensk energi) state that from a swedish perspective 1kWh corresponds to approximately 20g of carbon dioxide [92]. How much energy a laptop uses seems to vary quite a bit but around 250W for a good work laptop seems to be an worst case estimate. With those numbers an hour of prototyping in Zeppelin roughly corresponds to one sheet of paper. Not calculating the carbon print of manufacturing the computer since it is needed in the work as an programmer. So from an environmental point prototyping using Zeppelin is in most cases better for the environment than using paper prototypes.

Lastly some ethical aspects of this study. During the study the participants took part in the study on a voluntary basis and was aware and agreed to participate and that their responses would be anonymised and printed in this thesis. By recording and transcribing all interviews misinterpretations of what the participant said during an interview were min-
imised. The questions asked during the interviews were made as unbiased as possible to not influence the participant by asking weighted questions. In writing this thesis all sources have been interpreted and no plagiarism by copying and pasting text or images has been done.

### 6.6 Future work

When planning for and doing this study some choices were made that excluded possible alternatives. These alternatives are presented here as possible future work in the area. Instead of only choosing one tool to analyse it would be interesting to select several tools matching the criteria and compare them by doing the same tasks and evaluations in all of them to study if different tools are suited for different tasks. It would also be of interest to perform a case study of Zeppelin together with the customer to evaluate the usage of Zeppelin from their perspective since this study only focused on the developers perspective. Another improvement is that the selection and comparison of the visualisation tool could have been performed more in-depth to cover more tools and to analyse the tools matching the selection criteria more in-depth.

During the study some parts could have been done differently to perhaps get a more reliable result in this study. These improvements are listed below and could possibly give a more reliable result. To get evaluation data with better certainty a larger sample with more consultants in the evaluation could be used to get better reliability. If a more detailed selection of the participants in the evaluations could have been made any difference in knowledge of Python and SQL could be eliminated. Using other consultants for the second evaluation would remove any doubts regarding a learning curve. This would introduce other issues if the sample size is small since individual variances could affect the results.

While performing the study the question if the tool perhaps was better suited as a data analytics tool was brought up. In a tech-company that have lots of own data and some programming skills the use of Zeppelin as an analytics tool is not unreasonable. It would therefore be interesting to investigate if Zeppelin is suitable as a data analytics tool for a tech-company. Zeppelin would then have to be evaluated using other requirements than in this study. The size of data that can be analysed for example would probably have a larger impact if examined as a data analytics tool.
In this chapter each sub question from the research question will be answered and the answers to those questions will be used to answer the research question lastly in this chapter.

“Which tool is most suitable for using as a prototyping tool for data visualisations regarding Attentecs needs?”

From the results of the initial phase and using the criteria and looking at the tools that passed the criteria in this study, Apache Zeppelin is the most suitable tool for prototyping data visualisations based on information presented about the tools. This conclusion is solely based on the information displayed on the tools respective web page and on the tools found during the creation of the list. It is therefore possible that there exist a better tool for the task. Apache Zeppelin did however fulfil the criteria and based on the available information was the most suitable tool for using as a prototyping tool for data visualisations.

“Are there any limits in using the chosen tool for the case study for some characteristics or formats of data?”

To use Zeppelin data must be imported using an interpreter. There exist a number of interpreters for different programming languages such as Python and Apache Spark as mentioned in section 5.1.2 about Zeppelin. The limits on what data that can be used lays in the limits of the interpreter. As of the 11 of May 2017 no official support for MongoDB has been added to Zeppelin leaving the importing of data from MongoDB to other interpreters such as Python2 which have been used in this study for the import of the data.

Using the plug-in handler additional charts and interpreters can be added to Zeppelin for extended features. Between the 28 of March 2017 when the results from the initial phase, section 5.1.2 were written and the 11 of May 2017 eight additional visualisations have been added to the package handler indicating that charts for Attentecs needs likely will be created and added to the plug-in handler.

Zeppelin has no apparent limits in what data that can be used as long as it can be imported using an interpreter. When it comes to visualisations of data, the plug-in library will provide additional visualisations that are not already included in Zeppelin. At the time of choosing Apache Zeppelin 28 of March 2017 no geospatial visualisation was included As of 11 of May 2017 a geospatial visualisation is available in the plug-in handler but it is not yet functional.
“Is the chosen tool for the case study usable from a developer’s perspective on the tools perceived usability?”

The average SUS score was 70 in the first iteration and 68.5 in the second iteration which, according to Bangor et al. as described in section 2.6.2, is between acceptable and having some problems. From the interviews it were stated that Zeppelin have some issues regarding the text editor which might be one reason for not getting a higher score. Also, mentioned in the interviews is that the participants all think that Zeppelin is a usable tool. They all stated that they would use it again if presented with a task of visualising data, a couple of them with the addition that they thought that the tool was mostly suitable as a tool for prototyping visualisations and not as an end product. According to the SUS value and the interviews Apache Zeppelin is a usable tool from a developer’s perspective on the perceived usability although it has some problems and there exist room for improvements.

“Is it possible to make any improvements to the chosen tool to improve the perceived usability from a developer’s point of view?”

The SUS score did not change significantly between the two evaluations. The added features in implementation 2, section 5.5, did improve the overall SUM score for all but one task. Even if a learning curve is taken into account and that it is not insignificant, meaning that the hour of usage in the previous iteration affected the outcome, the tool shows either an increased usability or a very short learning period. Both an increased usability and a short learning period are good things. The changes to the second evaluation were also mentioned during the interviews as improvements of the usability which, to some degree, also indicates that the changes improved the perceived usability of Zeppelin. In summary the results from the second iteration on the changes made to Apache Zeppelin suggests that the changes improved the perceived usability of the tool from a developer’s point of view.

Using the answers to the above mentioned question it is now possible to answer the main research question.

“What are the limits and benefits with using a prototyping tool for aiding in the development of visualisations of data from Internet of things systems regarding data characteristics and the developers perceived usability of the system?” Zeppelin has no apparent limits in what data that can be used as long as it can be imported using an interpreter. When it comes to visualisations of data, the plug-in library will provide additional visualisations that is not already included in Zeppelin. According to the SUS value and the interviews Apache Zeppelin is a usable tool from a developer’s perspective on the perceived usability, but it has some problems and there exists room for improvements. The results from the second iteration on the changes made to Apache Zeppelin suggest that the changes improved the perceived usability of the tool from a developer’s point of view.
Bibliography


[50] H Saiedian and R Dale. “Requirements engineering: Making the connection between the software developer and customer”. In: *Information and Software Technology* 42.6 (2000), pp. 419–428. ISSN: 09505849. DOI: 10.1016/S0950-5849(99)00101-9


Appendix: Interview guides

A.1 Interview guide, pre-study regarding visualisation, with consultant

These interviews were held in Swedish and is therefore presented in Swedish.

Visualisering

- Kan du berätta lite om bakgrunden till visualiseringen?
  - Var det en del av ett större projekt?
  - Hur utvecklade ni visualiseringen i det projektet?
    * Visste kunden vad de ville ha?
    * Användes det någon prototyp?
  - Vilka typer av graf/grafer användes?
    * Fanns det några krav på visualiseringen? (Trendlinjer, gränser, etc.)

- Hur fick ni datan?
- Hur arbetade ni i projektet?
- Gjordes det några uppföljningar?
- Vad var svårest i projektet?
  - Att hämta in datan?
  - Att formatera datan?
  - Att skapa eller välja visualisering?
A.2 Interview guide, pre-study regarding IoT and work method, with managers

These interviews were held in Swedish and is therefore presented in Swedish.

Arbetssätt och kundinblanding i IoT projekt

• Hur ser en generell arbetsprocess ut i ett projekt?

• Finns det något som skiljer sig i arbetsprocessen inom ett IoT projekt?
  – Hur tas kraven på ett IoT system fram, mjukvarumässigt? (tillsammans med kund, kommer kunden med alla kraven, ändras dem under arbetet)
  – Tar ni fram någon prototyp? (papper, wireframe, färdigt simplifierat program)
    * Görs detta tidigt i arbetsprocessen?
    * Tillsammans med kunden eller gör ni den och sedan pratar med kunden om förändringar?

• Hur ser kundinblanding ut i övrigt i ett projekt?

• Vet kunden oftast exakt vad han eller hon vill ha eller behöver kundens tankar brytas ser i konkreta delar?

IoT-projekt

• Vad är den svåraste delen i implementationen av ett IoT system?

• Hur sker datainsamlingen? Om det skiljer i olika projekt kan du ge något eller några exempel?
  – Hur lagras datan?
  – Vad kom datan ifrån?
  – Hur hämtades datan?

• Hur såg datan ut?
  – Finns det några kännetecken på datan?
    * Formatet
    * Geodata
    * Trender
    * Cykler
    * Osäker data
    * Stora datamängder
    * Korrelationer
    * Övrigt
A.3 Interview guide, pre-study regarding IoT and work method, with consultant

These interviews were held in Swedish and is therefore presented in Swedish.

Arbetssätt och kundinblanding i IoT projekt

- Hur såg arbetsprocessen ut i ett projektet?
  - Hur togs kraven på systemet fram, mjukvarumässigt? (tillsammans med kund, kommer kunden med alla kraven, ändras dem under arbetet)
  - Tog ni fram någon prototyp? (papper, wireframe, färdigt simplifierat program)
    * Gjordes detta tidigt i arbetsprocessen?
    * Tillsammans med kunden eller gjorde ni den och sedan pratade med kunden om förändringar?

- Hur såg kundinblanding ut i övrigt i projektet?

- Visste kunden exakt vad de ville ha eller behövde kundens tankar brytas ser i konkreta delar?

IoT-projekt

- Vad var den svåraste delen i implementationen av projektet?
- Hur sker datainsamlingen?
  - Hur lagras datan?
  - Vad kom datan ifrån?
  - Hur hämtades datan?
- Hur såg datan ut?
  - Finns det några kännetecken på datan?
    * Formatet
    * Geodata
    * Trender
    * Cykler
    * Osäker data
    * Stora datamängder
    * Korrelationer
    * Övrigt
A.4 Interview guide, pre-study regarding IoT, visualisations and work method, with consultant

These interviews were held in Swedish and is therefore presented in Swedish.

Arbetssätt och kundinblanding i IoT projekt

- Hur såg arbetsprocessen ut i ett projektet?
  - Hur togs kraven på systemet fram, mjukvarumässigt? (tillsammans med kund, kommer kunden med alla kraven, ändras dem under arbetet)
  - Tog ni fram någon prototyp? (papper, wireframe, färdigt simplifierat program)
    - Gjordes detta tidigt i arbetsprocessen?
    - Tillsammans med kunden eller gjorde ni den och sedan pratade med kunden om förändringar?

- Hur såg kundinblanding ut i övrigt i projektet?
- Visste kunden exakt vad de ville ha eller behövde kundens tankar brytas ser i konkreta delar?

IoT-projekt

- Vad var den svåraste delen i implementationen av projektet?
- Hur sker datainsamlingen?
  - Hur lagras data?
  - Vad kom data ifrån?
  - Hur hämtades data?

- Hur såg data ut?

Visualisering

- Kan du berätta lite om bakgrunden till visualiseringen?
  - Var det en del av ett större projekt?
  - Hur utvecklade ni visualiseringen i det projektet?
    - Visste kunden vad de ville ha?
    - Användes det någon prototyp?
  - Vilka typer av graf/grafier användes?
    - Fanns det några krav på visualiseringen? (Trendlinjer, gränser, etc.)

- Hur fick ni data?
- Hur arbetade ni i projektet?
- Gjordes det några uppföljningar?
- Vad var svårast i projektet?
  - Att hämta in data?
  - Att formatera data?
  - Att skapa eller välja visualisering?
A.5 Interview guide, Intro to evaluation of a high-level prototyping tool for aiding in the development of visualisations

These interviews were held in Swedish and is therefore presented in Swedish.

- Vad har du för programmeringsbakgrund?
- Har du gjort någon form av datavisualisering tidigare?
- Är du bekväm i att programmera i Python?
- Har du grundläggande kunskap i att skriva SQL frågor?
A.6 Interview guide, evaluation of a high-level prototyping tool for aiding in the development of visualisations

These interviews were held in Swedish and is therefore presented in Swedish.

- Har du använt något liknande verktyg innan?
  - Vilket?
  - Skillnader?
  - Användbarast?

- Hur mycket påverkade din tidigare kunskap av Python möjligheten att genomföra den första uppgiften?

- Hur tyckte du att det gick?
- Vad var lättast och vad var svårast?
- Vad var det bästa med verktyget?
- Vad var det sämsta med verktyget?
- Finns det någon del man skulle kunna förenkla?
- Saknade du någon funktionalitet?

- Om du hade fått i uppdrag att lägga 40 timmar på att förbättra Zeppelin vad hade du gjort då?
- Om du fått i uppgift att genomföra den här uppgiften åt en kund, hur hade du angripit problemet då?
- Känner du att det tog längre tid att genomföra detta i Zeppelin än vad det hade gjort med något annat alternativ?
- Är Zeppelin från ditt perspektiv som utvecklare ett användbart verktyg?
- Skulle du använda Zeppelin igen?
  - Varför/Varför inte?
Appendix: Technical evaluation

B.1 Checklist

Data formats
- API data:
  - CSV
  - JSON
  - XML
- MySQL
- MongoDB

General limitations
- Memory
- Max number of results

Graphs
- Pixel-Oriented
- Geometric Projection
  - Scatter plots
  - Parallel Coordinates
  - Column charts
  - Line charts
  - Pie/doughnut chart
  - Geo-based charts/maps
• Icon-based

• Hierarchical
  – Treemaps
  – Sunburst graphs

• Graph-based or Visualising of complex data and relations
  – Tag clouds
  – Circular network diagram
  – Directed acyclic graphs

**Interaction**

• Selection
• Translation
• Rotation
• Filtering
• Zooming/scaling
• Linking and brushing
• Hotbox
• Gestures
Appendix: Evaluation tasks
C.1 Evaluation 1 of the usability of Apache Zeppelin.

These evaluations were held in Swedish and is therefore presented in Swedish. Utvärdering av användbarheten i Apache Zeppelin.

Din uppgift är att genomföra så många av uppgifterna nedan som möjligt. Uppgifterna kräver viss kunskap i Python och SQL. Om uppgift 1 tar mer än 30 minuter så kommer du att få tillgång en färdig version av den uppgiften för att kunna fortsätta med resterande uppgifter. Du får använda internet under utvärderingen om du skulle behöva googla något. Du får titta på och använda kod från mappen ”Example”.


Uppgifterna

1. Läs in datan från den lokala MongoDB databasen trafikverket med collectionen som heter weatherData formatera och mappa den till ett SQLContext som heter trafikverket_weather_data.
   - Id, Name, mättid, nordlig och östlig koordinat samt vägnummer ska alltid finnas med i trafikverket_weather_data, det är däremot inte säkert att de finns i weatherData. Om värdet inte finns ska värdet inte lagras.
   - Alla värden som innehåller Icon kan ignoreras. Övriga värden ska vara med om de finns lagrade.
   - Lagra nordlig och östlig koordinat separat från sweref99tm punkten.
   - Lagra avrundad mättid till närmsta halvtimme.


3. Visa nederbördsmängden från alla mätstationer mellan 2017-03-30 12:00:00 och 2017-03-30 12:31:00 på en karta.

4. Hämta vägtemperaturen från alla mätstationer från väg 22 mellan 2017-03-28 00:00:00 och 2017-04-04 00:00:00 och visa som linjediagram med en linje per mätstation.

5. Visa nederbördstypen och nederbördsmängden från alla mätstationer mellan 2017-03-30 12:00:00 och 2017-03-30 12:31:00 som punkter i ett koordinatsystem baserat på de geografiska koordinaterna.


8. Antal mätstationer per vägnummer som rapporterat mätvärden under senaste dygnet som histogram.
C.1. Evaluation 1 of the usability of Apache Zeppelin.

Exempeldata:

```json
{
  "Id": "SE_STA_VVIS703100",
  "Measurement": {
    "MeasureTime": "2017-03-27T16:10:02",
    "Precipitation": {
      "Amount": 3.4,
      "AmountName": "Måttligt regn",
      "Type": "Regn",
      "TypeIconId": "precipitationRain"
    },
    "Road": {
      "Temp": 14.8,
      "TempIconId": "tempAirRoad"
    },
    "Air": {
      "RelativeHumidity": 74,
      "Temp": 8.8,
      "TempIconId": "tempAirRoad"
    },
    "Wind": {
      "Direction": 270,
      "DirectionText": "Väst",
      "DirectionIconId": "windW",
      "Force": 6.8,
      "ForceMax": 13.4
    }
  },
  "Name": "703_Laganrasten_Fjärryta",
  "Geometry": {
    "SWEREF99TM": "POINT(438907.99 6311209.89)"
  },
  "RoadNumberNumeric": 4
}
```

Tack för att du deltar i denna utvärdering av användbarheten i Apache Zeppelin.
C.2 Evaluation 2 of the usability of Apache Zeppelin.

These evaluations were held in Swedish and is therefore presented in Swedish.

Utvärdering av användbarheten i Apache Zeppelin.

Din uppgift är att genomföra så många av uppgifterna nedan som möjligt. Uppgifterna kräver viss kunskap i Python och SQL. Om uppgift 1 tar mer än 30 minuter så kommer du att få tillgång en färdig version av den uppgiften för att kunna fortsätta med resterande uppgifter. Du får använda internet under utvärderingen om du skulle behöva googla något. Du får titta på och använda kod från mappen ’Example’.


Uppgifterna

1. Läs in datan från den lokala MongoDB databasen trafikverket med collectionen som heter trainData formatera och mappa den till ett SQLContext som heter trainData.
   - Följande fält ska finnas med i importen ActivityId, ActivityType, AdvertisedTimeAtLocation, AdvertisedTrainId, Canceled, Endpoint, AdvertisedLocationName, InformationOwner, Location, AdvertisedLocationName, Location, Geometry, SWEREF99TM, OtherInformation, ProductInformation, ScheduledDepartureDateTime, StartStation, AdvertisedLocationName, TimeAtLocation, TypeOfTraffic
   - Om data för ett av fälten inte finns kan det sättas till null.
   - SWEREF99TM positioner ska delas upp i en nordlig och en östlig del.
   - En tidsdifferans mellan TimeAtLocation och AdvertisedTimeAtLocation ska lagras i minuter. Om inte både TimeAtLocation och AdvertisedTimeAtLocation finns ska databasraden inte sparas.

2. Hämta all data från en station under en tidsperiod, baserat på den annonserade tiden för platsen, och visa upp denna data i form av en tabell. Stationen och mellan vilka datum och tidpunkter som mätvärdena ska hämtas ska gå att ändra med textfält. Använd station Linköping C mellan 2017-04-07 04:00:00 och 2017-04-07 08:00:00 som defaultvärde.

3. Visa den genomsnittliga förseningen vid stationerna för alla tåg mellan 2017-04-07 06:00:00 och 2017-04-07 07:00:00 på en karta.

4. Hämta förseningarna från Stockholm Central mellan 2017-04-07 00:00:00 och 2017-04-09 00:00:00 och visa dessa i ett linjediagram.

5. Visa förseningen och slutstation från alla stationer mellan 2017-04-07 07:00:00 och 2017-04-07 08:00:00 som punkter i ett koordinatsystem baserat på de geografiska koordinaterna på en karta. Färgen ska indikera slutstationen och storleken förseningen.

6. Visualisera den genomsnittliga förseningen för de 20 stationer med högst genomsnittlig försening mellan 2017-04-07 15:00:00 och 2017-04-07 19:00:00, med ett cirkeldiagram. Använd GROUP BY i SQL queryn.

7. Visa genomsnittlig försening för tågnummer 8713 på en karta.
8. Visa genomsnittlig försening från Stockholm Central per företag, informationsägare, som histogram.
C.2. Evaluation 2 of the usability of Apache Zeppelin.

Exempeldata:

```json
{
   "ActivityId": "2500adde-05b5-4f00-08d4-71899df0a6cd",
   "ActivityType": "Ankomst",
   "Advertised": true,
   "AdvertisedTimeAtLocation": "2017-04-07T22:11:00",
   "AdvertisedTrainIdent": "1113",
   "Canceled": false,
   "EndStation": {
      "AdvertisedLocationName": "Köpenhamn H",
      "AdvertisedShortLocationName": "Köpenhamn H",
      "Geometry": {
         "SWEREF99TM": "POINT(347093,6172713)"
      },
      "LocationSignature": "Dk.kh"
   },
   "EstimatedTimeIsPreliminary": false,
   "InformationOwner": "Öresundståg",
   "Location": {
      "AdvertisedLocationName": "Malmö C",
      "AdvertisedShortLocationName": "Malmö",
      "Geometry": {
         "SWEREF99TM": "POINT(374034,6164458)"
      },
      "LocationSignature": "M"
   },
   "LocationSignature": "M",
   "ModifiedTime": "2017-04-07T23:05:55.235Z",
   "NextStation": {
      "AdvertisedLocationName": "Köpenhamn H",
      "AdvertisedShortLocationName": "Köpenhamn H",
      "Geometry": {
         "SWEREF99TM": "POINT(347093,6172713)"
      },
      "LocationSignature": "Dk.kh"
   },
   "OtherInformation": "Slutstation Österport",
   "PreviousStation": {
      "AdvertisedLocationName": "Eslöv",
      "AdvertisedShortLocationName": "Eslöv",
      "Geometry": {
         "SWEREF99TM": "POINT(393873,6189313)"
      },
      "LocationSignature": "E"
   },
   "ProductInformation": "Öresundståg",
   "ScheduledDepartureDateTime": "2017-04-07T00:00:00",
   "StartStation": {
      "AdvertisedLocationName": "Kalmar C",
      "AdvertisedShortLocationName": "Kalmar",
      "Geometry": {
         "SWEREF99TM": "POINT(583383,6280513)"
      }
   }
}``
C.2. Evaluation 2 of the usability of Apache Zeppelin.

Tack för att du deltar i denna utvärdering av användbarheten i Apache Zeppelin.
Appendix: Evaluation, post task questionnaire
## D.1 Evaluation, post task questionnaire

These evaluations were held in Swedish and is therefore presented in Swedish. Utvärderings frågeformulär att genomföra efter varje slutförd uppgift.

Uppgifts nummer:

Deltagaren namn:

| Hur skulle du beskriva hur svårt eller lätt det var att slutföra uppgiften? |
|-------------------------------------------------|-----------------|
| Väldigt svårt | 1 | 2 | 3 | 4 | 5 | Väldigt lätt |

| Hur nöjd är du med att använda det här programmet för att slutföra uppgiften? |
|-------------------------------------------------|-----------------|
| Väldigt missnöjd | 1 | 2 | 3 | 4 | 5 | Väldigt nöjd |

| Hur skulle du bedöma hur lång tid det tog att slutföra denna uppgift? |
|-------------------------------------------------|-----------------|
| För mycket tid | 1 | 2 | 3 | 4 | 5 | Väldigt lite tid |
Appendix: Evaluation, SUS questionnaire
E.1 Evaluation, SUS questionnaire

These evaluations were held in Swedish and is therefore presented in Swedish.

**SUS utvärderingsformulär att genomföra efter slutförd evaluatoring.**

Deltagaren namn:

<table>
<thead>
<tr>
<th>Jag tror att jag skulle vilja använda systemet frekvent.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag tycker systemet var onödigt komplex.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag tycker att systemet var lätt att använda.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag tror att jag skulle behöva stöd av en senior anändare för att kunna använda systemet.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag finner att de olika funktionerna i system var väl integrerade.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag tycker att det var för mycket inkonsekvent i systemet.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag kan tänka mig att de flesta människor skulle lära sig att använda systemet väldigt snabbt.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag finner systemet mycket besvärliga att använda.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag kände mig väldigt säker i att använda systemet.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Jag behövde lära sig en massa saker innan jag kunde komma igång med systemet.</th>
<th>Instämmer inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Instämmer starkt</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>