Development of a Project Planning Support tool with User Centered Design

Nguyen Lam and Denhi Huynh

DEPARTMENT OF DESIGN SCIENCES
LUND UNIVERSITY
2016

MASTER THESIS

Development of a project planning support tool with user centered design

Nguyen Lam and Denhi Huynh

Development of a planning support tool with user centered design

Published by
Department of Design Sciences
Faculty of Engineering LTH, Lund University
P.O. Box 118, SE-221 00 Lund, Sweden

Subject: Interaction Design (MAMM01)
Supervisor: Joakim Eriksson
Examiner: Kirsten Rassmus-Gröhn

Sammanfattning

Idag är planering en essentiell komponent i både den offentliga och den privata sektorn. Ett bra planerad projekt har en högre sannolikhet för att lyckas än ett inte så välplanerad projekt. Men ju större ett projekt är, desto mer tidskrävande är planeringsprocessen.

I detta examensarbete utvecklades ett planeringsverktyg för att hitta en tillräcklig bra konfiguration mellan personer och deluppgifter som utgör ett projekt, baserad på personers kompetens och deluppgifters krav. För att lösa detta implementerades och visualiserades två algoritmer, en för att lösa den kritiska vägen och en genetisk algoritm. Den kritiska vägen algoritmen räknar ut vilka deluppgifter som förskjuter slutdatumet för projektet och den genetiska användes för att hitta en bra konfiguration, när man allokerar personer till en deluppgift.

Visualiseringar skapades som extensions till Qliks mjukvara Qlik Sense. Dessa extensions var designad med en användarcentrerad designmetod, där förbättringar och vidareutveckling var baserade på användarnas återkoppling. In de tidigare stadier skapades prototyper för att få en överblick. Vid slutfaserna adderades interaktion till prototyperna för att därefter testas på användarna. Dessa användare var en blandning av nybörjare och expertanvändare av Qlik Sense. Deltagarna från användarstudierna tyckte att prototypen var interaktivt, men många föreslog även nya funktionalitet som önskades. Den slutgiltiga prototypen bestod av två extensions, där projekt managers kan få en överblick of alla deluppgifter och motsvarigheten av personerna som ska utföra dessa deluppgifter.

Slutligen, måste en del arbete genomföras för att släppa ut en slutversion. Prototypen kan även modifieras för att tillämpas inom andra branscher.

Abstract

Today, planning is an essential part in most lines of business. A good planned project has a higher chance of succeeding than a not so well planned project. However, the bigger a project is, the more time consuming the planning phase becomes.

This master thesis has been aimed at designing and developing a planning support tool for finding a good match between persons and tasks based on their skill set and skill requirements respectively. Two algorithms were implemented and visualized in this project, the critical path algorithm and the genetic algorithm. The critical path algorithm calculates the tasks which postpones the end date of the project and the genetic algorithm was used to find a good configuration when assigning a person to a task.

The visualizations were created as extensions to Qlik's software Qlik sense. The extensions were designed with a user centered design, where improvements and further development was based on users feedback. In the early stages, prototypes were created. In the later stages, interaction with the extension was added and then tested on users. The test users were a mixture of beginners and advanced users of Qlik sense. From the user study, it was noted that the users have no problem to interact with the prototype, but many of them suggested features that they desired. The final software created was two extensions, where a project manager can get an overview of all tasks and the corresponding person assigned to execute the task.

It can be concluded that further work would be necessary in order to create a release ready version of the tool. The tool also has to be modified to suit other lines of businesses.

Keywords

Project management, scheduling, planning, Gantt chart, task assignment, critical path, genetic algorithm, user centered design, brainstorming, lo-fi prototyping, hi-fi prototyping, user study.

Acknowledgement

We would like to thank Qliktech International AB for allowing us to conduct our master thesis at their company. We want to thank our supervisor at Qlik, Quan Ho, who have been a huge source of support and wisdom. We would also like to thank Christoffer Hedberg and Imir Useini, who's been helping us will all kinds of questions regarding Qlik sense and other implementation details.

A huge thanks goes out to all managers at Qlik, who has participated in our interviews and also all participants for our user study. Their feedback has been of utmost importance and has been used as a base for the development of the planning support tool.

A huge thank also goes to our supervisor at Lund university, Joakim Eriksson, who with his support and knowledge, has helped us the entire way of this project.

We would also like to thank our two master thesis neighbors at Qlik, Karl Jansson and Simon Alm Liljevall for their invaluable input and effort of solving mutual problems that occurred in our projects.

Contents

1 Introduction

Planning has always been crucial to the human being, whether planning for everyday life or long-term planning. Ever since the beginning of mankind, humans have been able to plan, using different strategies to stalk out a prey in groups and thereafter planning an attack to capture a target. Our ability to hunt in groups and coordinating attacks are the main differences between us and other predators.

Today, planning is used in various projects across all sectors of business and the reason for this is that projects are complicated and comprises of non-repeatable tasks with an approximated deadline. Therefore, the planning within an organization will require a lot of effort and this could possibly cause delays to projects. However, planning well could result in savings such as time, resources and capital.

When planning for software project, the project manager has to take into account the skill set of each developer and try to match them with the tasks in the software project. Today, this task assignment is usually performed manually, and to consider all candidates for each task is very time consuming.

The question is, could this be automated and how would the results be visualized?

1.1 About Qlik

This master thesis is made in collaboration with Qliktech international AB (below referred to as Qlik). At Qlik, intuitive business intelligence solutions are created for self-service data visualization, guided analytics applications, embedded analytics and reporting. Qlik is a company founded by Björn Berg and Staffan Gestrelius year 1993 in Lund, Sweden. Their vision was to create software which mimics the way the brain works. One key feature which has become iconic for the company is the color coding created by Björn Berg. In his color coding, selected values are highlighted in green, linked values were highlighted in white and excluded values were shaded out in gray [1].

Today, Qlik has more than 36000 customers spread worldwide. Qlik also has more than 2000 employees situated in offices around the world. Qlik's headquarter is situated in Radnor, Pennsylvania, in the United states. However, most development still takes place in Lund, Sweden. Qlik's two most successful applications are QlikView and Qlik sense. In this project, it will be investigated if it is possible to integrate a planning support tool with the Qlik sense software.

Qlikview is one of the products created by Qlik. The QlikView software was created in 1996, and was designed to give the users an incredibly detailed data analysis using just a single click, and thus the name QlikView [1].

Traditional data visualization tools aimed to create graphs based on some predefined questions. The problem with this approach was that it was difficult for the regular user to ask follow up questions. This means that in order for the users of the tool to get this new information they had to ask the IT-department to make some new queries. This was very time consuming and a tedious process in order to gain some data in order to create data visualizations. Qlik Sense is a product which aimed to solve this problem.

Qlik sense offers a user friendly drag-and-drop software in order to create self service data visualizations. Using Qlik sense, users could now create their own visualizations in a simple and intuitive manner. Qlik sense gives instant feedback in all graphs related to the users queries based on the user's own given input. This completely removes the middle stage where the user had to ask an IT department in order to create a new query, and the user could now create multiple queries in almost no time.

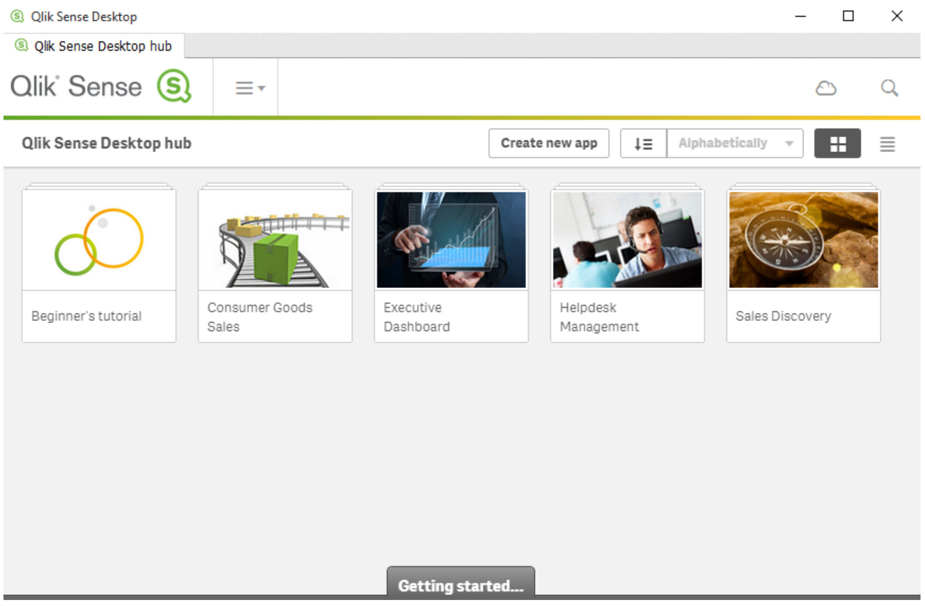


Figure 1: Start screen in Qlik sense with five pre-created apps.

The Qlik sense software allows users to create their own "apps". In figure 1 the start screen of the Qlik sense software is shown. In the start screen, five different apps are shown, namely the beginners tutorial, consumer goods sales, executive dashboard, help desk management and sales discovery. The user could also create their own apps for visualizing their own data.

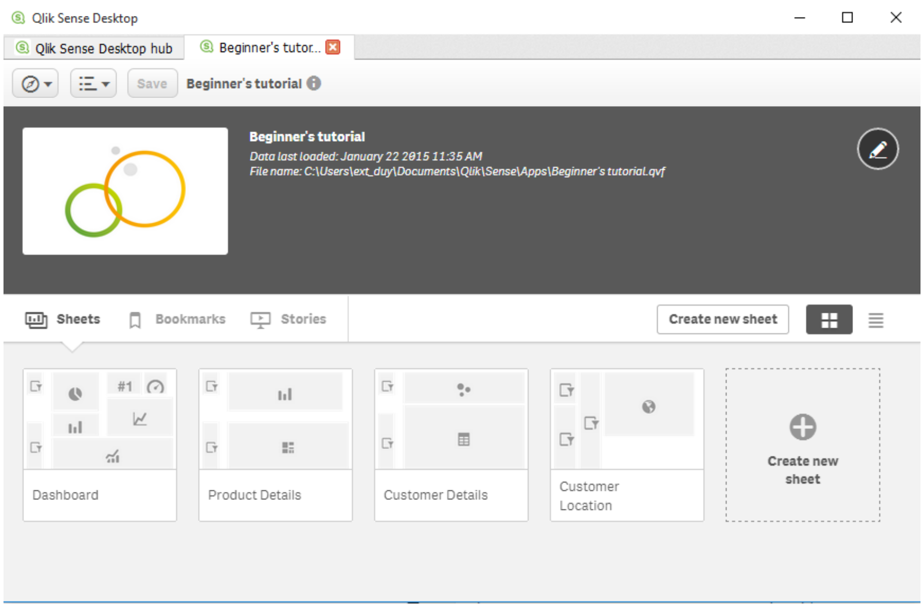


Figure 2: Sheets inside an app.

In each app, the user could add "sheets" which is a collection of different graphs. In figure 2, the app "Beginners tutorial" with four created sheets are shown. The sheets in this example is dashboard, product details, customer details, and customer location.

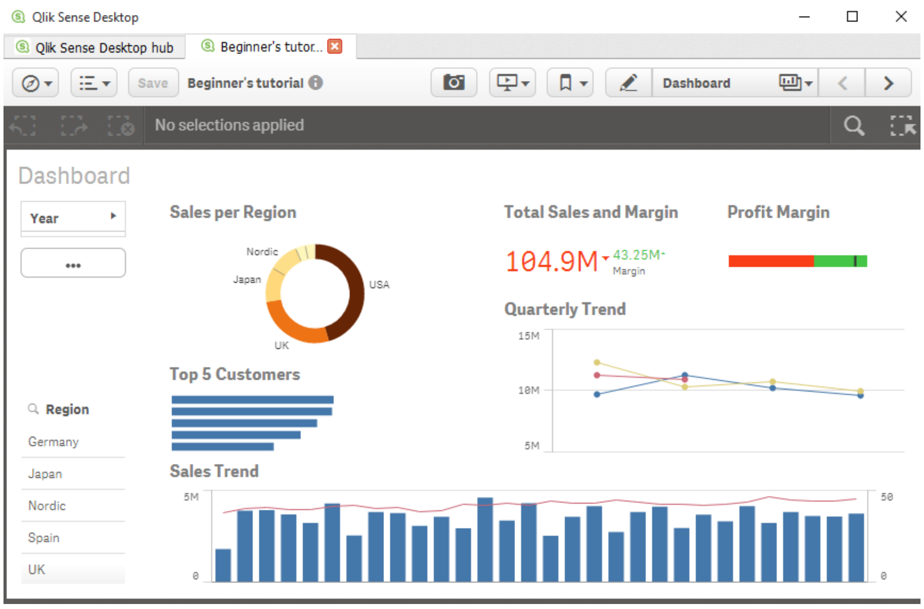


Figure 3: Graphs inside a sheet.

Each sheet has a collection of different graphs, which the user can choose by themselves. In figure 3 there is a collection of six graphs with the title sales per region, top 5 customers, sales trend, total sales and margins, profit margins and quarterly trend. In the sheet, the user has the possibility to filter the graphs using different inputs. For example the user could choose a specific country in order to see the sales in that country. The different graphs available (in version 2.1.1 + Build:22) is the pie chart, bar charts, combo charts, kpi, gauge, line chart. To get a better image of how everything works, one could download the software for free at Qlik's website [2].

1.2 Purpose

In this master thesis, the objective is to investigate and create an interactive visualization tool prototype in order to support project planning within an organization. The intended end user of the software is primarily project managers, but could also involve other stakeholders as well. The goal is to make the prototype as generic as possible in order to address all sectors of business. However, the focus will be on software management. A typical scenario for managers is to assign a person to a task, given that the person fulfills the requirements, while also accounting for any upcoming delays and dependencies of tasks that might occur.

1.3 Problem formulation

Given a set of resources and tasks, the objectives were to find:

- A good configuration of assigning tasks to resources, such that some given constraints and objectives are fulfilled.
- An adequate visualization which can be intepreted in such a way that the user will find the prototype intuitive and usable.

What a resource is depends on which line of business a company is in. Resources are needed in order to complete the different tasks found in a project. A task is defined as some objective which have measurable estimates of performance criteria such as task duration, cost, and resource consumption. Furthermore, a task can require one or several resources and have one or several operating modes. For example, a task in a sewing factory can be to produce a certain number of t-shirts, which could be performed in a manual operating mode using a person powered sewing machine or completely in automatic mode using a mass producing machine.

Constraints in a project are goals which must be completely fulfilled in order to execute the project. For example, a project can not be finished if a task requires resources which aren't available at the given time of execution.

Objectives are different goals which one wants to maximize or minimize in a project. Some objectives can be to minimize the makespan time(the duration of the project), minimize costs, maximize match rate between resource and tasks or maximize the workload balance between different resources. Different objectives can contradict each other.

1.4 Scope

This report consist of three main parts, the theoretical background, the process, and discussion. The theoretical background consists of terms and definitions used in design, the software development cycle and in scheduling. The process is our development process and design process combined presented in a chronological order (oldest first). Lastly, the discussion consists of our reasoning during the process and what and why decisions were made.

Given the many variations of resources and task modes, a smaller subset of problems had to be chosen for this master thesis. The subset chosen was scheduling for software projects.

In software projects, the resources are employees, which are renewable resources. Each resource has a skill set, and each skill has a skill level which is a measurement of how well the employee can utilize the given skill. For example in software projects the skill set can consists of both development process methodologies or knowledge in different programming languages. The tasks has only one mode of execution, which in software projects means that a programmer develops software. A software project consists of many tasks. The definition of task used in this project is that a task is a problem which can be solved using a single resource, i.e. one employee per task. The duration of the task is the time for one individual to complete the task working full time.

For this planning support tool, no constraints are given. The objectives for the scheduling are to maximize a balanced workload and to maximize the match rates between task requirements and the skill sets of the resources. Furthermore, the tasks will be tried to be scheduled such that the resources are available at the given time interval assigned for the task.

For the visualization part, user centered design will be used as a design process to develop the extension. As a development process, the scrum method will be used. Both concepts will be explained under the theoretical background section.

2 Theoretical background

2.1 Designing interactive products

When designing a product, there is not one fundamental design approach one can follow. However, there exists several different concepts or methods which one could utilize when designing software. One can combine different design concepts or methods into a so called "design process". In this section, an introduction to different concepts within user-centered design considered applicable for this project, are explained. In addition, the key word "product", refers to both general products and software products.

2.1.1 Data collection via interviews and surveys

Data collection needs to performed in order to provide insight regarding the features of the project. This can be done in a qualitative approach(interviews) and in a quantitative approach(surveys).

Interviews are composed of open questions, which allow the participants to answer freely. However, in order to extract valuable information, the interviewer must use keywords such as "tell", "explain" and "why". The keyword "why" is needed to explore the depth of a certain individual's opinions in contrast to "tell" and "explain", which are less investigative keywords. In interviews, it is of utmost importance to gain some clarity of user experiences of the participant and to ensure that the participant expresses their opinions of a certain event.

Interviews are good for conducting in depth analysis and provides a lot of data for the prototype, however interviews are time consuming in regards to the actual interview, but also synchronization takes time. For example, meetings must be booked and although a personal meeting is preferable, it is not always possible. For further notice, closed questions should always be avoided as these questions are reserved for surveys [3].

Surveys are used to provide quantitative data and are used in conjunction with closed questions such as binary questions or relative questions. Binary questions contains only two alternatives as answers such as true or false and yes or no [3]. Relative questions could be constructed based on a scale. For example, a question could be: Do you think that the current visualizations of the program is intuitive? After the questions, the participants are able to give it a rating between 1 and 5, where 1 is unsatisfactory and 5 is satisfactory.

2.1.2 Brainstorming

Brainstorming is an method for producing as many ideas as possible within a short time frame. An important aspect of brainstorming is to not criticize any opinions and this in turn could generate unfeasible ideas, however such approaches are encouraged in brainstorming.

The process of brainstorming starts out with team taking notes of individual ideas. Then, the ideas are read aloud and for each idea, the team share their thoughts, with the exception of not being allowed to criticize any idea. Finally, the ideas are categorized according to their similarities and relations [3].

Another more structured variant of brainstorming is method 635, in which six participants are hand-picked, write down three ideas individually on a piece of paper, then the paper is handed to the nearest neighbor and the neighbor will write three additional ideas or modify the current existing ideas. This process is repeated until the piece of paper returns to it's original author. This method avoids the existing problem of a single idea taking over the brainstorming session and it is more possible to associate an idea to a participant, however the intensity and productivity(in quantity) is lost, if method 635 is used instead of usual brainstorming [3].

2.1.3 Prototyping

A low fidelity prototype (lo-fi prototype) is a low detail, fast and cheap prototype created by sketches and usually written on paper. The disadvantage to lo-fi prototyping is that interactions can be hard to show on sketches. The advantage of lo-fi prototyping is that one is able to visualize and design a simple prototype without spending that much time or money. Therefore, decisions can be made very quickly and different designs can be evaluated and one can move on with the project faster compared to having a detailed prototype or implementing actual code [3].

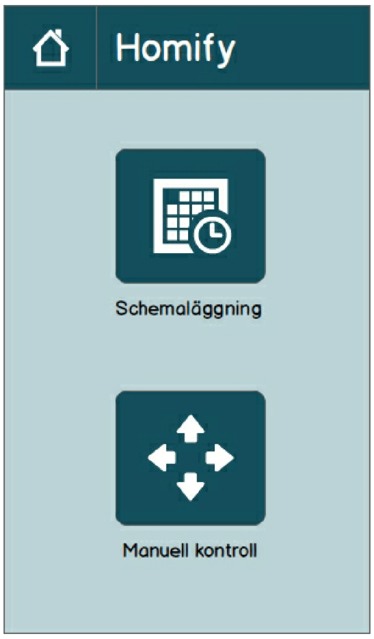


Figure 4: A hi-fi prototype describing the front page of an app for an automatic vacuum cleaner.

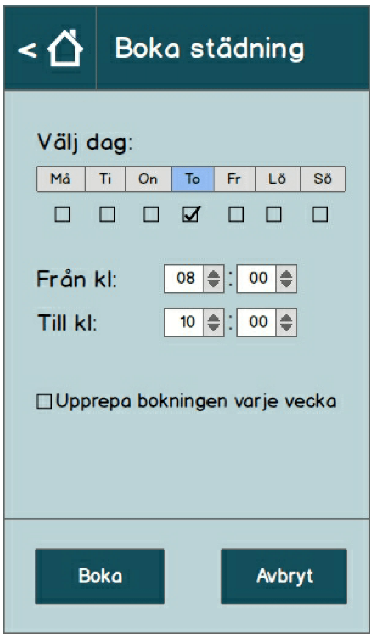


Figure 5: A hi-fi prototype describing how to schedule a cleaning on a specific day and time.

A high fidelity prototype (hi-fi prototype) is a high detail prototype usually created using mock-up (model) programs or other computer software. The disadvantage to a hi-fi prototype is that it is not as flexible and fast as a lo-fi prototype. However, it is faster to create a hi-fi prototype than the actual software. The advantage to a hi-fi prototype is that it is easy to test user interaction and check if the product is as intuitive as the designer thinks or hopes. Usually both lo-fi prototyping and hi-fi prototyping are used during the design process [3].

A horizontal prototype is a prototype which has many function but each function is not very detailed. A horizontal prototype can be used to display an all round scope which is to be included in a product. The purpose of the horizontal prototype is to clarify and explore alternative solutions.

The opposite to a horizontal prototype is called a vertical prototype, which has a small set of functions but each function is very detailed. A vertical prototype is used to focus on a small area and evaluate the design for the given functions in this subset of total functions. The purpose of vertical prototyping is to determine if a feature or solution is adequate before implementing a large scale solution [4].

Another prototyping method is to let a user interact with a product, but have another person control the product's behavior. This method is known as the "The wizard of Oz", referring to the wizard controlling a fake wizard and its behavior [3].

Evolutionary prototyping is when a product is developed from the start. Then instead of discarding different prototypes, one instead build upon this prototype. The purpose of evolutionary prototypes is to gradually adapt the product to changes which might not be possible to determine in early stages of the design process [4].

2.1.4 Usability Study

In order to evaluate the resulting prototype, one could run usability tests to confirm that a user knows how to interact with the product and to see if one's assumptions about the interaction with the product are accurate. The first action is to address high-level questions that will be needed to complete the study. An example would be: Is the study going to be a benchmark to evaluate usability for a certain product? This is known as study goals. In addition, the users goals are metrics that defines the user experience such as their performance and their satisfaction. By combining the results from the study goals and the user goals, one can find the appropriate metrics to use when analyzing the outcome of the study [5].

Before commencing the study, it needs to be planned and the organizers of the study needs to decide how to use the data gathered from the participants, There are two approaches to perform this. The first approach is called formative usability, which means that usability experts evaluates the usability and improves upon the design iteratively. Then the tests are carried out and the cycle is repeated again. Formative usability addresses issues such as what prevents the user from performing a task, failure and errors that occurs during the tests.

On the contrary, summative usability focuses on evaluating the design after it is ready and there is a prototype to be tested on. The objective in this case is to check if the resulting prototype meets certain predefined criteria. Questions that arises during this phase are for example, how well the prototype fulfill the usability goals or if improvements regarding the usability has been made comparing to the earlier release of the product.

The user goals of the product can be different depending on the targeted user group. It can depend on the frequency of usage for the product, the user might demand that the product must be efficient in run time etc. However, there are two metrics, when measuring user goals, which are performance and satisfaction. Performance involves measuring all actions performed by the user. For example, one can measure, the amount of time users spends on each task, the amount of errors users make and the time it takes for the user to learn the system. Satisfaction involves all opinions that the user have regarding the product. For example, the user may mention that the system was easy to use, hard to use, intuitive or non-intuitive. Study goals combined with user goals will be the foundations for continuing the study.

For a usability test, a small amount of test users are required. According to an article by Jakob Nielsen, that number is five [6]. The reasoning behind this is that more people than five are usually not required in order to find the problem areas of a product. Even if the usability test has more people, they will most likely have the same problems. For a company, the return of investment(ROI) is not much higher for including more test users. One should instead conduct a lot of different test with smaller groups instead. Nielsen claims that the accumulated benefit of running several small tests is better than the benefit of running one big test with more test users.

To rephrase, a small number of test users are suitable for formative studies [5]. The approach starts out with five test users identifying major usability issues with the product. Then the design is improved upon and another five test users are selected to evaluate the new design. The cycle continues until a consensus is reached upon the final design before the actual release. However, according to Tullis and Albert, it is advisable to select a larger sample set for summative studies in order to have low discrepancy and be able to generalize the data to draw statistical conclusion regarding the result. In summative user studies, the goal is to test if any small design modifications have any impact on the usability of the product and in order to perform that properly, it is generally a good approach to have more than 100 participants.

During the actual test, the participant is seated at the computer with the software already launched. Near the participants are one moderator and one annotator. The moderator will ask the participant to perform the tasks and observe the result while the annotator will write down the results such as the amount of time spent on a particular task or any opinions that the participant have on the product. It is of utmost importance that the annotator write down the data according to an agreed pattern in order to ease the analysis process.

How the moderator should moderate depends on what the end goal is. A few different moderating techniques are concurrent think aloud (CTA), retrospective think aloud (RTA), concurrent probing (CP) and retrospective probing (RP).

In concurrent think aloud method, the participant is asked to tell out loud what the participant is thinking when trying to perform a task. This give hints on where improvements need to be made if the participant works in a way not expected by the developers. The con about this method is that some usability metrics based on time will be inaccurate since extra time is taken by the participant to explain its actions.

In retrospective think aloud, the participant is asked to retrace its working pattern after the test session has finished. However, if the user study is long, then the participant might not be able to recall its steps.

In concurrent probing, follow up questions are asked when the participant performs a unique action or says something unique. Usage of concurrent probing might however disrupt the line of thought that the participant has during a user study session.

Lastly, retrospective probing is when the follow up questions are asked after the user study session has finished. Retrospective probing has the same problem as retrospective think aloud, the participant might not recall its actions after the session.

2.1.5 Metaphors

Metaphors are used to map real life phenomenon with abstract data, in order to simplify the data for the human brain. A good example is the traffic light metaphor. In a traditional traffic light system, there exists green, yellow and red light. The green light is commonly associated with keywords such as "safe", "clear", "approved". Yellow light usually indicates warnings and finally, red light indicates danger.

For this masters thesis project, metaphors will be investigated in conjunction to visualizations, in other words, given different metaphors, will the metaphors yield different interpretations of the same visual information? According to the article by Ziemkiewicz and Kosara [7], the authors asked questions regarding visualizations. The participants were given either a tree-map or a node link structure and thereafter given four questions related to their graph and four questions unrelated to their graph. For example a participant assigned to the tree-map were given four questions related to the tree-map and four questions related to the node link structure. The result of this study is that the participants responded more slowly to incompatible questions than questions related to the given graph. Furthermore, the compatible questions was not only answered faster, but the accuracy was better as well. This study, highlights the importance of metaphors, needing to be constructed as accurate as possible in order to interpret the graph correctly. In addition, using metaphors could map cognitive thinking to a visual representation in a powerful way if applied correctly, but this can backfire if the metaphor can not be associated with the provided visualizations.

2.1.6 Affordance

When developing a product, it is important to guide the user to which actions are available. The properties of a product determines the approaches the user will take in order to interact with the product. This is commonly referred to as affordance. For example, a computer comes with predefined affordance, the keyboard invites a user to interact with the keys and the mouse has a scrolling wheel and buttons which invites the user to click and scroll the mouse.

There is both perceived affordance and real affordance. Perceived affordance is how the user thinks it is supposed to interact with the product and real affordance is how a user is actually supposed to interact with a product. However, Don Norman argues that all instances of "affordance" or "real affordance" should be changed to "perceived affordance" since in the end, its how a user perceived how to use your product which is important, not how it is actually supposed to work [8].

2.1.7 Constraints

Closely related to affordance is constraints. Constraints refers to preventing a user interacting in certain ways with a product. Don Norman has made a distinction of three different kinds of constraints, physical, logical and cultural constraints.

Physical constraint is a constraint given by the physical limitations of the product. For example, a user can not move the mouse pointer outside of the screen (unless the operating system allows the mouse pointer to "go outside the screen").

A logical constraint is when a user has to deduce which further actions could be taken by analyzing which information is in front of the user. For example, if a survey has the question "Which of the following ten candy types do you prefer the most?", but only seven of the alternatives are shown on the screen, the user can logically deduce that there must be another three options which are not showing on the screen and the user therefore has to scroll down in order to see those three missing options.

A cultural constraint is a learned behavior of how a user is supposed to interact with a product. One example of a cultural constraint is that the user has to click the scroll-bar on the right side and drag it downwards in order to move the page upwards. This is based on convention and there could have been a lot of other equivalently good options on how to scroll a page (for example holding both the right and left button to drag the page in a certain direction) [8].

2.1.8 Conceptual and mental models

A conceptual model is a highly simplified explanation of how something works. A conceptual model is used simplify the usage of product, it doesn't have to be accurate, but should be useful. For example, in order to help people create a conceptual model, computers has folders, icons and files. Although that is not how the computer actually stores data (it is all in the memory of the computer), it simplifies the usage of the computer for users by hiding the abstraction of how the computer actually works [9].

A mental model are the conceptual models of the people's mind representing their understanding of how something works. People can have different mental models of the same product or item. In fact, even the same person can have different mental models of how a certain item works. The same person can have different mental models for different aspects of the some item, they could even contradict each other.

It is usually enough to just know the cause and effect of most items. For example, most people understand how to use a switch to turn on a lamp, but few knows the actual underlying mechanics behind it. As long as an item is used as expected then there is no need for a good mental model. However, it is when an item does not work as expected where a good mental model matters. Then, it could be the difference between fixing the item and not.

2.1.9 Feedback

Feedback is about the clarity of the consequences of an action made by the user. A feedback is good if the user is aware of the actions made by themselves. This can be accomplished by providing a connectivity between different screens, web pages and modes. This process is called visual momentum and this can be hard to achieve, due to that the visualizations created must display the general structure of the data, in order for the user to not keep too much information in their head. There has to exists landmarks to inform the user of their location and which navigation is possible from that specific location [3].

Furthermore, feedback is required to occur immediately, as a slightest delay is even noticeable to a user. For example, if one presses a button and nothing happens, the person might repress prompting two delayed events, which can be annoying to the user.

2.1.10 Feed forward

In contrast, to feedback, feed forward explores the next available actions for the user. An example could be that the user enters an elevator and are presented with a lot of options. The user can choose to close the door by pressing the button indicated with inward arrows, keep the door open by pressing the button with the outward arrows or press a number button to travel to another floor. All these actions are some kind of advances that the user can make. It is suggested that flashing lights can be used to give the user an option performing that particular action [10].

2.1.11 Mapping

Mapping means the relationship between elements from two sets of things [9]. Mapping is an important property to have in mind when designing a product. A product with a good mapping uses a good spatial correspondence between the two sets of elements. This is known as natural mapping. Natural mapping means that spatial analogies are used in order to map between the elements of the two sets. According to Donald Norman [9], a natural mapping leads to immediate understanding.

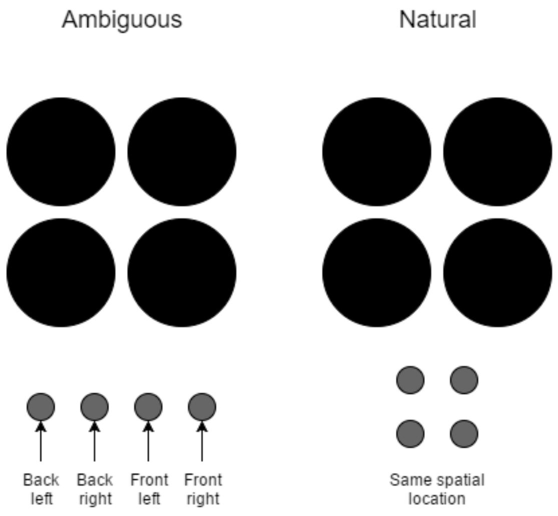


Figure 6: Mapping of buttons (gray) to a stove (black).

An example of a good mapping is the steering wheels found in cars, when the user rotates it to the left the car turns to the left and when the user rotates the wheel to the right the car turns to the right. A bad mapping could be when a stove has two plates lying vertically one behind the other but the buttons are placed horizontally (as the left stove in figure 6), which button should then control the upper stove plate and the lower one? The natural mapping as in the stove to the right in figure 6 is much better.

2.1.12 Discoverability

Discoverability is an important characteristic of a product. A good product should have good discoverability, meaning that the product should be designed in a way that the user is able to tell which actions are possible at any time.

2.2 Development

One important aspect to developing a product is to have an effective work flow. One solution is to use a development model. There exist several different methods, both with their pros and cons. There exists both sequential development models and also iterative development models. In a sequential development model one step at a time is executed until the product is finished. Contrary to sequential development models, is iterative models. In iterative development models, each step is performed in small sprints, where a few features are developed each time. Iterative models are based on the idea of evolutionary prototyping. A few different methods will be explained below.

2.2.1 The waterfall method

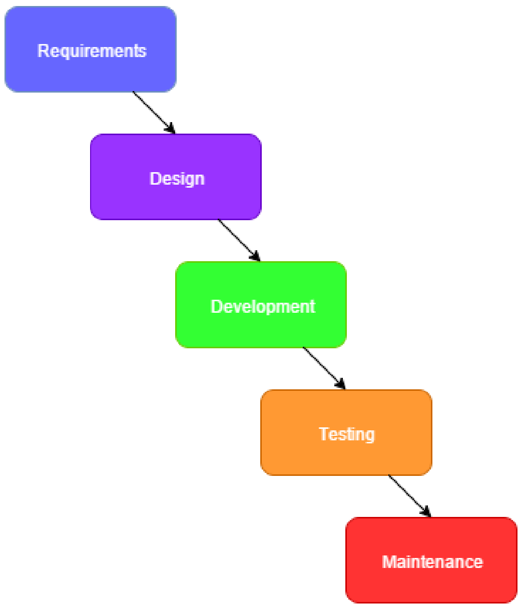


Figure 7: Waterfall development model.

The waterfall method belongs to the category of sequential development models. This means that one step in the model is performed at the time, and the next step does not begin before the previous step is finished. There are a few different variants of the waterfall model, but almost all have the five stages seen in figure 7 above.In the waterfall model, all requirements are decided first. When all the requirements of the system are in place, the design step takes place. When the design foundation has been laid, it is time to start the development. After development, testing is performed to ensure that the software works appropriately. Finally, the last step is maintenance of the product [11].

The advantage of using the waterfall model is that all requirements are laid down before developing any code. Therefore, the implementation step might be faster compared to not having any direct requirements to follow. However, the waterfall method has one huge disadvantage, a client might change a requirement or add a new one depending on the need. This could result in a lot of code having to be refactored, changed, or deleted.

2.2.2 Extreme programming

Extreme programming is an iterative development model. In extreme programming, small iterations of analysis, design, implementation and tests are performed. One iteration usually vary between two weeks and two months. Some companies might even go with one week iterations, but usually not above two months iteration [3].

Using an iterative method such as extreme programming, the developers do not plan so far into the future, which allows the development process to be very flexible with what features to include and if a client have more requested features [12].

Extreme programming or XP follows a set of practices. Each practice is supposed to work well together with the other practices. In the beginning, the development team together with the customer has to analyze which features should be included in the coming iteration. Each features is called a "story". It could take a month to plan stories for development for one entire year. Then the development team estimate how long each story will take and how important they are to the product. Based on the estimates given from the programmers, the customer or customers decide a time plan and scope for the project [12].

Instead of one big release, extreme programming model suggest many small releases but often. The first release does not even have to solve the main problem it aims to solve. The releases could be as often as daily or monthly. Metaphors are used to define the shape of the program and is shared between the programmer and the client. Extreme programming promotes following the DRY-method, which means don't repeat yourself. It means that do not use duplicate code, instead define it only once and use it everywhere [12].

Tests are a big part of extreme programming. Unit tests are written very often and are used to check if the code works as expected. Customers also write functional tests in order to check if a function of the program works as expected [12].

In extreme programming, the code base is often refactored in order to improve readability and enhance productivity. Refactoring means that transformations are made to the code (e.g. variable name changes, extraction of methods) which does not affect the behaviour of the program and the end result should still make all tests running successfully [12].

In the development step, pair programming is used. This means that the developers are not developing by themselves. Instead, as the name suggest, they develop in pairs. This allows the programmers to easily catch the mistakes made by the other programmer which is controlling the computer [12].

The code is continually integrated into the working software. If the tests does not pass, the changes are discarded. Also, if the programmer finds an error somewhere in the code, they are allowed to change it, no matter where its written or who wrote it [12].

Of course, the above written practices are just guidelines and each developer team can decide for themselves which to include and which to exclude.

2.2.3 SCRUM

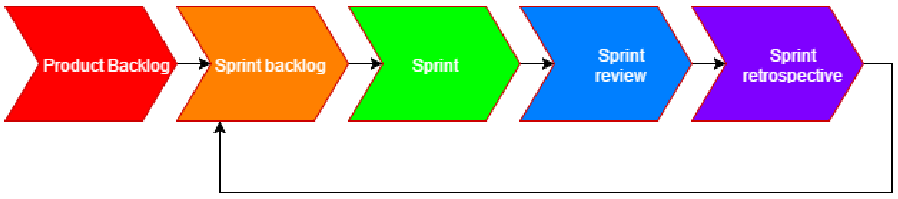


Figure 8: SCRUM development model.

SCRUM is another iterative development process. Figure 8 show a basic version of SCRUM, where each part will be described below. In SCRUM, there are key persons, the product owner, the development team and the scrum master. Their roles will be explained as their roles become essential in the SCRUM development cycle.

Before iterating, the product owner creates a product backlog, which is a list of desired features for the product. The product owner represents the customer and its stakeholders.

In each iteration (or sprint), the product owner creates a sprint backlog, which is a list of features that should be implemented and included in the coming sprint. An alternative to having the product owner decide the scope of the sprint is to have a sprint planning event, where the developers can decide which backlog items could be implemented. The sprint has a predefined duration (i.e. it is time boxed). Usually one sprint takes between one week and one month with two weeks being the most common.

When it is decided what features should be in the sprint, it is time for the sprint. During this phase the developers try to implement the features of the sprint backlog.

Everyday during the sprint, a short stand up (or daily SCRUM) meeting is held, where the developers discuss what they did yesterday which contributed to the project and what each person will do today.

During the entire SCRUM process, the SCRUM master makes sure that the SCRUM principles are followed and that the project scope and time is met.

After the sprint phase, a sprint review is performed. In the sprint review, the development team meet the product owner and reviews features that was implemented and features that was not.

Lastly in each sprint, the sprint retrospective phase is performed. In this phase, the development team reflects on the last sprint and discusses what actions could be taken to further improve the development process [13] [14].

2.3 Scheduling

2.3.1 Tasks

A project consists of several tasks, which in this paper is defined as small objectives to be solved by one person. A task has properties such as a duration, skill requirements and dependencies. The properties for the task must be decided before using the task assignment tool.

The duration is the estimated time of completion for the task. For tasks in product companies, the duration can sometimes be known beforehand. For example, the duration in order to produce a given product might be known based on data about the machine or perhaps empirically generated(by using the machine and take the time). For software projects, there is no definitive way of deciding the duration, therefore an estimate must be made. The estimate is usually based on previous knowledge and compared to similar problems solved in the past.

The skill requirements is given in a list where each skill has a name and a skill level. For software projects, the skills can be knowledge in a given programming language, where each level represents how well the developer knows the programming language, and perhaps different technologies using that programming language. The scale of the skills can be decided arbitrarily. For example, skill level 1 could mean that the programmer knows the fundamental language syntax, and higher levels could mean that the developer knows how to utilize this language in order to perform specific tasks such as communication with networks, or creating graphical user interfaces in the language. The skills and the skill levels has to be decided beforehand. Note that although this thesis focuses on software projects, the skills can be arbitrary and the skill levels as well.

Tasks may have dependencies. This means that some tasks can not be executed before some other tasks has finished or partly finished. In addition, there are also stricter dependencies called critical path. Critical path is a set of tasks for which there must be no delays, otherwise the entire project will be delayed. In a software project, tasks including planning the system structure has to be made before continuing the implementation of the different parts of the system structure.

2.3.2 Resources

In order to complete a task, some kind of resource has to be used. In software development, the resources are software developers and of course computers. In other lines of businesses the resources can be a machine or raw materials.

In this paper the resources are people. Each person has a skill set consisting of skills with a certain skill level, exactly the same as the tasks.

2.3.3 Constraints

Constraints are goals which must be 100% fulfilled in order to perform a given task.

In software projects, this means that the software developer must be available at the given time of execution of the task. This constraint can be violated by a user having vacation days or perhaps by a worker getting sick. Other constraints can be that all tasks must be finished before the deadline.

In other businesses, constraints can be that a certain amount of resources must exists in order to execute a given task.

2.3.4 Objectives

Objectives are goals, which one wants to fulfill as good as possible, but it is not necessary to fulfill them 100%. Sometimes the objectives can not be fulfilled perfectly. Some objectives may even contradict each other.

In software development scheduling, one wants the maximize the match rate for each person assigned to a task. If this was the only requirement, then the best developer would have to perform all tasks. Of course this is not sustainable solution since one developer can only work so much in the given time span of the work. Therefore, another objective could be to balance the workload as much as possible. A third objective could be to minimize the makespan of the project, which is the total duration of the project. This is usually very important for the employer since time costs money. Having multiple objectives means that there must be some compromises made. For example, to get a more load balanced schedule, perhaps a person with a less good matchrate has to be assigned to some given task.

2.3.5 Algorithms

In order to distinguish between a good and a bad task assignment solution, one could compare the fitness score. The fitness score is a measurement which measure how well the solution fulfills the given constraints and objectives. For a smaller data set, a brute-force algorithm which tries all possible combinations and compares fitness score might work. However when the problem grows in size, this approach is no longer applicable. Imagine for example a problem which has 100 tasks, and for each task one of 100 software developers should be assigned. Even if we only allow each person to perform one task (in contrast to being able to select each employee several times), the first task can be assigned to 100 different software developers, the next one could then be assigned to 99 different software developers and the next 98 and so one. In total this is 100!(factorial) different combinations and to try each combination using a regular computer is unfeasible. Now note that the real problem is bigger than that, since each person should be able to be assigned to multiple tasks. A brute-force solution would have to try 100^{100} different combinations for this problem. Therefore to solve this task assignment problem, other approaches had to be tried. For this thesis a heuristic algorithm and an genetic algorithm was tried.

A heuristic algorithm does not try all possible solutions as a brute-force algorithm does. Instead, a heuristic algorithm follows rules of thumb, an educated guess. The goal of a heuristic method is not to find the best possible solution but to generate a solution which is "good enough". What "good enough" means depends on the problem one wants to solve. In order to find good heuristics, it is important that the developer have certain insight in the given problem field. The advantage of a heuristic algorithm is that it finds a solution very fast. However a heuristic algorithm might not take all requirements into account. Also, given many different objectives it is difficult to find good heuristics which satisfies all objectives equally. Instead it might optimize for some criteria more than other [15].

A genetic algorithm mimics the natural evolution process in order to solve a problem. A genetic algorithm finds multiple(a population of) solutions at once, and then the set of solution is combined in order to create new solutions. Which of these solutions that survive each generation depends on their fitness score. Note that a genetic algorithm is not only specifically used for task assignment problems, but the general idea can be applied for solving the task assignment problem [16].

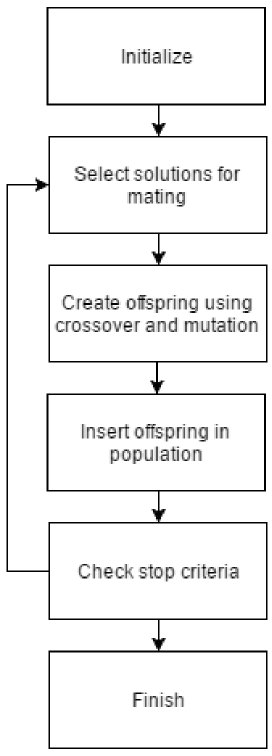

Initialize
Select solutions for mating
Create offspring using crossover and mutation
Insert offspring in population
Check stop criteria
Finish

Figure 9: Basic structure of a genetic algorithm.

Figure 9 shows the basic structure of a genetic algorithm [16].

In the initialization step a population of different solutions is generated. What a solution looks like depends on the problem one wants to solve. For the task assignment problem a solution might be a list containing which person that should perform which task.

After the initialization step two of the solutions are selected for mating. Which solutions that are selected could vary depending on the the fitness of the solution.

When two solutions have been selected the mating begins. The mating consist of two different steps, the crossover step and the mutation step. When creating the offspring, either one or two children can be created.

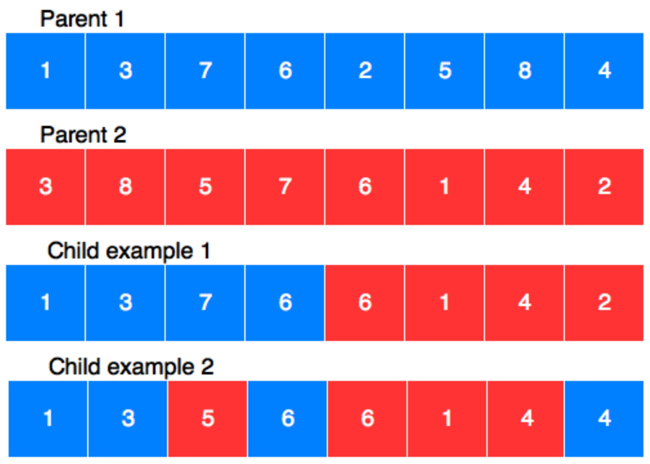


Figure 10: Two different ways of performing crossover.

During the crossover step one or two offspring solutions are created by taking parts from the parents and combine them into a new solution. There are multiple ways of perform the crossover. One way is to take the starting part from the first solution and the second part from the other solution and the reversed (figure 10, child example 1). Another way is to randomly choose from one or the other of the parent solutions for each part of the solution (figure 10, child example 1). For the task assignment problem the latter was used, where for each task, one of the parent solution's person was chosen randomly.

During the mutation step the solutions have a chance for mutation. What specific part of a solution can mutate to, can be implemented in different ways. For some problems, a normally distributed mutation can be used. For other problems, it is enough to randomly mutate to any other value existing in the problem set. For the task assignment, that means that one of the assigned task might mutate and not be assigned from one of the given parent values.

When a new population of offspring has been created, then some or all of them could be put back into the population. In this step the solutions which has a higher fitness score have a higher probability to be chosen and inserted into the population. Of course, as there are new solutions inserted into the population size will grow. Therefore some or all solutions from the parent generation are removed. Which to remove of course also depends how well the solution solves the problem.

In order to calculate a solution's fitness, some measurements must be used. The measurements used are heuristics, meaning that what they are depends on which problem one wants to solve. A fitness function consist of two parts, a constraint score and an objective score. The constraint score depends on how well a solution fulfills the constraints, which ideally is 100 percent. If the constraint score corresponds to 100 percent in some measurement, then the objective score should be calculated. The objective score depends on how well the objectives are fulfilled. The objective score should be as high as possible, but it is not certain a solution with 100% exists. Objectives could contradict each other, meaning that in order to get a higher score on one of the objectives, there might be some trade-off made with another objective.

Now that the new generation of solution has been created, it is time to check the stopping criteria. If the stopping criteria has not been fulfilled, the cycle of creating offspring is run again. But if it has been found, the algorithm is finished.

The stopping criteria could be decided in several ways. One way is to have a fitness threshold, which means that the algorithm is finished when there is at least one solution which fulfills the fitness threshold. The problem with this approach is that due to the fact that the objective score might not even have a 100% solution, it is hard to set a threshold which one might be sure that a solution can fulfill. Another approach is to check when the fitness score has converged to a specific number. When the best solution has converged to a specific number, it does not mean that the best possible solution has been found. However, at least a local maximum has been found. In addition to the convergence criteria, one could also set a maximum number of generations before ending the algorithm.

3 Method

In this section methods for the development and design process is presented. This chapter describes the whole process, which is interviews, brainstorming session, five sprints for development and design and lastly user tests.

3.1 Work flow

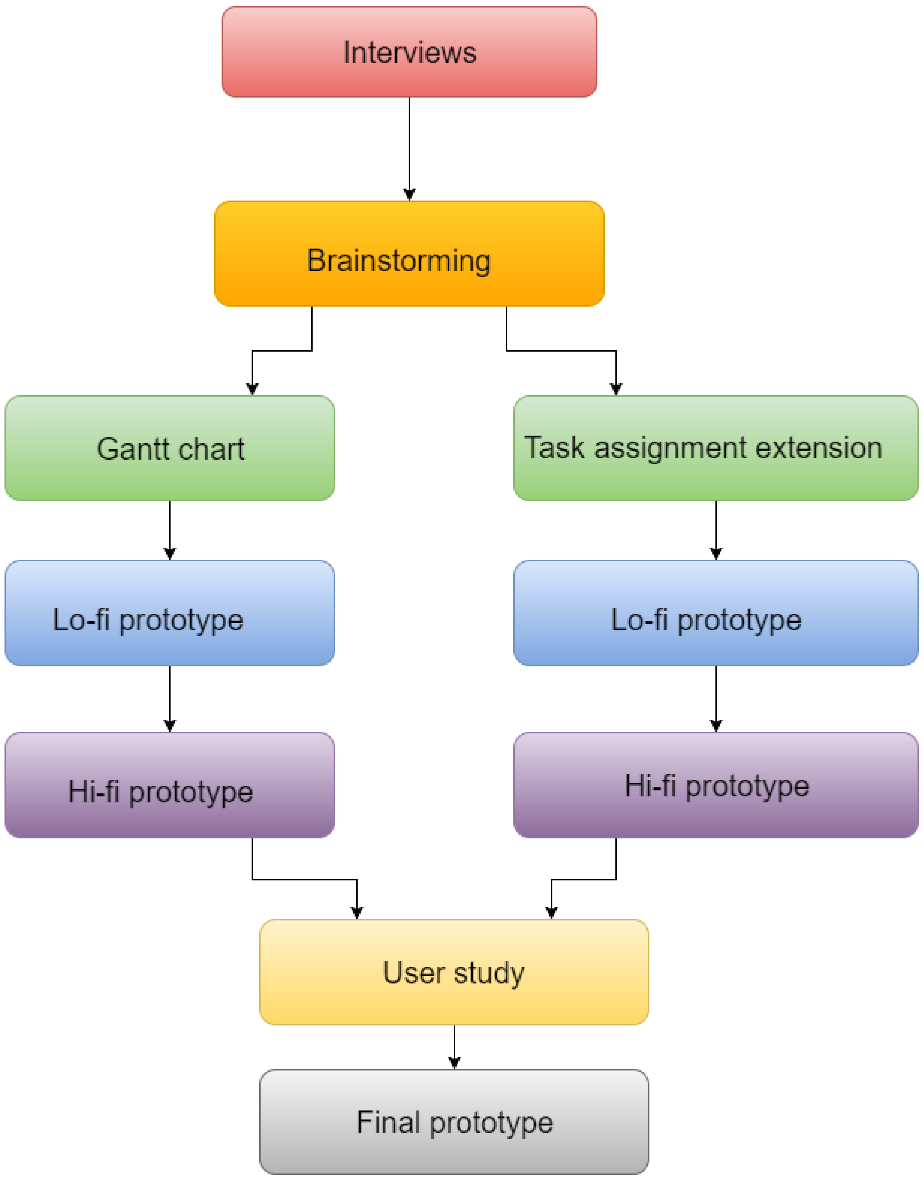


Figure 11: Describes the work flow during the entire project

In figure 11, the work flow of the process comprising the whole project can be observed. A single arrow indicates that much work was made together while the double arrows separating brainstorming with Task assignment and Gantt chart indicates work that was done in parallel.

When developing this planning support tool, an iterative development model was used. The iterative development model was a combination of scrum and some concepts from extreme programming. However, this does not mean that the waterfall model is disregarded, but rather that the model is of lower importance. During each sprint in the scrum process a design process was succeeded by the implementation phase, continuing with the test phase and ending with the review phase. The entire development process is described below in chronological order.

3.2 Investigation and conceptual design

3.2.1 Interviews

Before starting the design work, our task was get a good understanding of how and what planning was performed. Interviews with managers at Qlik were held in order to get a good initial understanding of which information was searched for by project managers and also to get a look into the flow of information during the planning phase. The questions for the interviews were first written individually, in order to get a wider span of questions, followed by a discussion to choose the most appropriate questions.

The interviews was based on asking the participants open questions and let them answer freely regarding their experiences with software management. During the interviews, some spontaneous follow-up questions not included in the questions sheet were asked. Furthermore, notes were taken to reflect the opinions of the participants and simple diagrams were made to further aid the comprehension. Lastly, the data from the interviews will be used as a basis to create a lo-fi prototype.

3.2.2 Brainstorming

The brainstorming session was used to generate many ideas within a short time frame and started out by 15 minutes of individual brainstorming, where associations for the prototype were created. During this process, no restrictions were imposed on the ideas and the ideas could take any form, whether they were crazy, funny, creative etc. After the initial 15 minutes, the remaining time was used to discuss the ideas and to merge the ideas altogether.

3.2.3 Lo-fi prototype

Based on the results from the brainstorming session, a lo-fi prototype to display the tasks was created and the approach for constructing it was made on A3 paper with coloured pencils. Furthermore, it was decided that the tasks were to be displayed in a Gantt-chart, which is a type of bar chart. The Gantt-chart displays the tasks on a timeline horizontally. It was created by Karol Adamiecki in the mid 1890s and later adapted by Henry Gantt. A Gantt-chart shows information such as the task start date, task end date, overlapping tasks and project start- and end date [17]. A Gantt chart was used because it shows all important aspects of the task, such as start date, end date and duration. One could also see dependencies between the different tasks.

3.2.4 Hi-fi prototype

For the hi-fi prototype, in the initial phase a decision was made to just include the most basic parts of the lo-fi prototype in order to not have everything during the first sprint. It was decided that for the first iteration, the projects should be visualized and a specific task or project should be selectable. All other interactions were left for a later sprints.

3.3 Sprint 1

3.3.1 Task assignment extension

The first part was to start developing the algorithm for matching tasks and personnel. The algorithm part is a task which will last more than one sprint, since it's a work in progress during the entire scope of this project. Before starting the work on the algorithm, a data format for the visualization had to be decided.

During this sprint, it was discovered that Qlik Sense is better used to visualize data, and not for data manipulation. Therefore some of the earlier ideas such as adjusting the duration of the task by dragging them or move the tasks were dismissed.

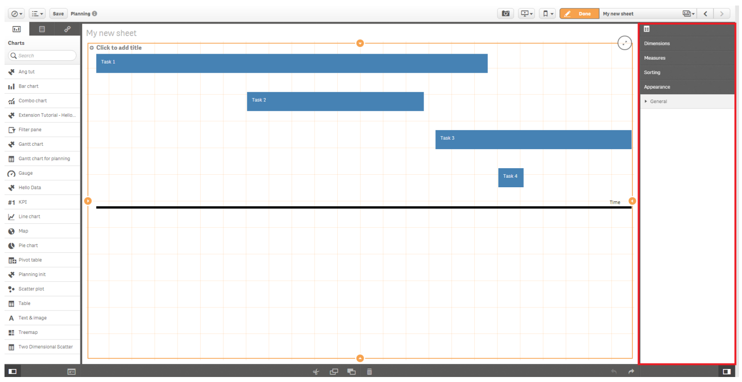


Figure 12: Properties panel inside Qlik Sense tool marked in red.

After deciding which format to use, the Qlik sense API had to be learned in order to figure out how to integrate the given data into the extension. For this part, built in Qlik Sense API and Qlik Sense guidelines were used. It was decided that the user had to input the data in the properties panel inside Qlik Sense, which is shown in figure 12.

Up to this point, the focus had only been on creating the Gantt-chart for displaying the different tasks. However, during this sprint, a first version for a task assignment algorithm was created. It was decided that the result from the task assignment algorithm would be visualized in another extension, and not in the Gantt-chart extension. For the task assignment problem, the objectives were to minimize the makespan, maximize the match rate and maximize the workload balance.

In order to find a "good enough" solution, our first idea was based on the idea of critical path. Calculating for the critical path will yield the best possible makespan of the project.

The initial thought was to first calculate the critical path of the project, then make sure that the personnel with as good match rate as possible was assigned to those tasks. The people with less good match rate could then be assigned to the other tasks not included in the critical path where there was extra time, which could be used in order to learn the new skills required in order to solve the problem.

3.3.2 Gantt chart

The second part of the work was to develop the user interface such as described by the hi-fi prototype. This interface is created by using a JavaScript library called D3(Data-Driven Documents) [19] and libraries for extension developers in Qlik Sense. D3 is a library which allows the developer to create powerful and interactive visualizations.

3.4 Sprint 2

3.4.1 Task assignment extension

During this sprint, it was decided that an attempt to merge the heuristic solution with a visualization were to be made. Simultaneously, a new task assignment algorithm approach was tried.

The problem with the previous approach was that makespan and match rate were preferred over a balanced workload for the employees. Instead, scheduling using a genetic algorithm approach was tested, which became the final approach used to solve this problem. During sprint 2 an initial version for this algorithm was created. The initial version had two objectives, maximize the match rate between the task and persons and to maximize the workload balance.

3.5 Gantt chart

At the beginning of this sprint, the visualization was still a work in progress in this step. However, while the user interface yielded from sprint one provided some basic visualization, it has to be improved, in order to provide empirical data. For example, the chart shows the length of each task, but does not highlight any relation between the tasks. Furthermore, the time line does not show any numbers(see figure 12) and these changes were to be adjusted for sprint two.

3.6 Sprint 3

3.6.1 Task assignment extension

During sprint 3 it was discussed if further work on the algorithm were to made or if a visualization for the task assignment, distinct from the Gantt-chart, were to be created. Due to the uncertainty of how much of the implementation would be finished in the scope of this master thesis it felt more realistic to create the visualization instead of continuing on the algorithm, although the expectation is to finish both.

Since creating the visualization for the task assignment was decided, this sprint had another design process. During this design process, the goal was to develop an extension to visualize which person that was assigned to which task, and also to be able to see which tasks each person had been assigned to.

Since a genetic algorithm was used another objective was to show the corresponding data. There are two relations between tasks and persons. Each person can have one or many tasks, while each task can only be assigned to one person.

3.6.2 Gantt chart

In parallel to the genetic algorithm, the development of the Gantt chart representing the tasks and it's dependencies continued. During this iteration, it was decided that the visualization representing the timeline for the Gantt chart should be moved to the top center of the canvas(The canvas is the space in which the bars reside in). This will expand the dimensions of the canvas, reserving space for more bars. The method for achieving this effect is to apply a bigger canvas, exceeding the dimension of the computer screen, on the smaller canvas comprising the original dimensions. In addition, this activates a scroll bar on the right hand side of the screen, which enables the user to scroll down in order to observe the other bars.

3.7 Sprint 4

3.7.1 Task assignment extension

In sprint 4 the zoom in feature was removed and filtering added instead. When implementing this new feature, there was no longer any need to have the first layer of the partition layout, since it was there to allow a user to zoom out. Therefore, the first layer was removed for the extension.

In addition to filtering, solid colors were added to the extension by using a color scheme adapted by Google, which does not take into consideration lighter colors such as lightgrey, lightblue, lightgreen and so forth.

3.7.2 Gantt chart

To date, the Gantt chart has not been tested on a bigger data input and the objective of this iteration was to test the limitations of the Gantt chart. In addition, selecting a subset of the elements and confirming this selection makes the visualization invalid due to previous calculation not obeying the laws of division. This resulted in the dependencies being removed temporarily until a solution arises and this particular problem was resolved during this iteration. In addition, the Gantt chart extension has also been integrated with the task assignment extension and the goal of the integration is to use both extensions in order to aid the data discovery.

Hitherto, there was no possibility to observe the present time and compare the time available until the deadline of the project. This is temporarily solved by implementing a line, crossing over all tasks vertically in order to highlight the current date. The line is placed relative to the total duration of the entire span of the project. For example, if the total duration of the project is 40 days and the screen width is 1920 units, then each day corresponds to 48 units. Let's say for the sake of simplicity that the current date is on day 23. Then it is located at the position 1104(23*48). In a Cartesian coordinate system the start point is at (1104, 0). This means that the x-coordinate is at 1104 and the y-coordinate is at 0(at the top of the canvas).

3.8 Sprint 5

3.8.1 Task assignment extension

For the task assignment extension, not much visual changes were made in sprint 5. The mouse cursor was changed to a hand instead of an arrow on in order to provide affordance that the extensions were clickable. Furthermore, on hovering above a person to task assignment, the currently active choice in the task assignment have a black edge around it.

Functionality to actually perform the selections were added. For example, when a person or some persons has been selected, only the relevant data for those persons were to be displayed in all extensions. In this case, the extensions that needed to be updated was the Gantt-chart and the task assignment algorithm.

Since there now was a working task assignment extension, it was decided to try and improve the algorithm again. This time, an overlapping score was added. The overlapping score was an objective which measured how much the assigned task for a person was overlapping in the schedule.

An important thing to note, is that the overlap score does not guarantee that there will be no overlaps, only that it will optimize for it. Since the duration of a task was defined as the estimated time for one full time worker to finish it, it is preferable to have as little overlap as possible.

3.8.2 Gantt chart

In this sprint, the dependency lines were modified to a step function between the start and the end node. A step function is a straight line, which can turn either left or right once at the end of path. In addition, a legend was added to clarify the distinction between a non-critical path and a critical path. For this visualization, a black arrow line represent a non critical path between two tasks and a red arrow line represent a critical path between two tasks. The orange vertical line were implemented to represent the current day relative to the timeline at the top of the canvas. For each interval on the timeline, there are vertical gray lines and these are supposed to be used as a reference for the user to identify the start date and end date of an arbitrary task. As for the task assignment, a function was added such that, when hovering over the bars, the mouse cursor changes to an almost to a closed fist with an upward pointing index finger and the bars gets a black edge around it.

3.9 Usability testing

Before sprint 5, the task assignment extension and the Gantt chart extension were in a too early stage to conduct usability tests. After sprint 5, the software was working to the extent that usability tests could be held. The formative approach was chosen as the method for testing and it begins by selecting a few participants to identify major design flaws. These flaws were fixed and then a new sample set was selected to test the new design. This process is then iterated until the developers are satisfied with the results.The formative approach in conjunction with ”concurrent think aloud” method was used for the usability testing in order to follow the thought process of the user.

The usability testing has several questions designed to let the user explore all the possibilities of interaction for the visualization(See the appendix for the actual questions). The questions are split into two parts. The first part consists of questions that lets the user interact with the prototype. These questions are asked as openly as possible, in other words most questions are quite short. The second part are feedback questions regarding the subjective opinions of the participant. The entire process is monitored by authors, where one is asking the questions and observing the user's behavior patterns and the other writes down notes regarding the performance and opinions of the user.

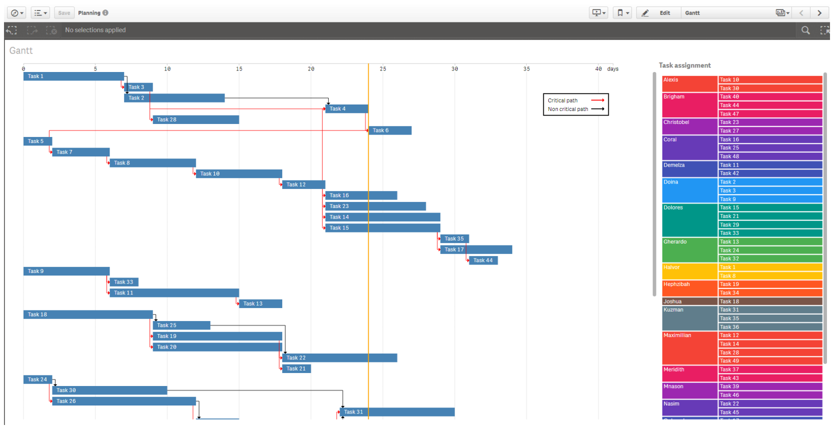


Figure 13: First design to be tested.

Figure 13 shows the first design to be tested. For this iteration of the usability study, there was 4 test users in total. Two of the test users had previous experience with Qlik sense and other two had almost no experience with Qlik sense. In total, the usability test had 11 test scenarios and 3 retrospective questions for the test users to answer.

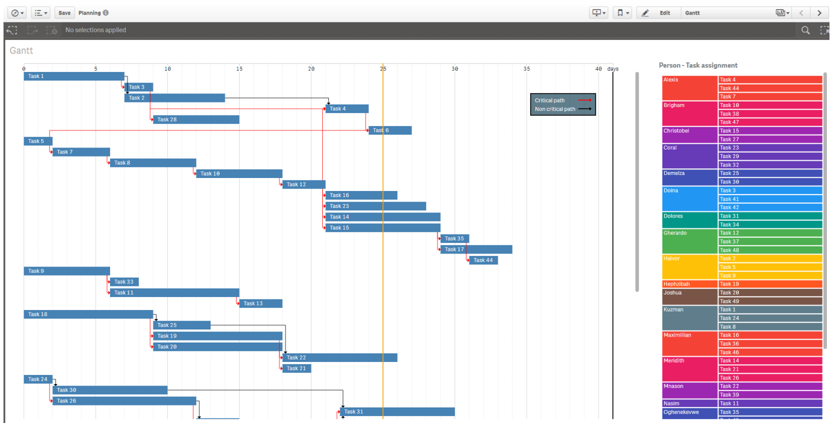


Figure 14: Second design to be tested.

From the given feedback, improvements were made before conducting the next test. The legend and today text were adjusted such that they are always displayed at their relative position even when scrolling down. Furthermore, small lines were added for the day to day basis in order to improve readability of the start and end dates of the tasks. The legend background color was also adjusted in order to improve visibility.

After the improvements, five participants were selected for further usability testing and their objective was to interact with the visualization displayed in figure 14. The difference for this is that all the participants are working at Qlik and are accustomed to the product.

4 Result

Here the results of the different parts of the project will be presented.

4.1 Investigation and conceptual design

4.1.1 Interviews

At the infancy of the interviews, five project managers were selected to participate. From the interviews, one thing all managers agreed upon was that time estimation was regarded difficult and they spoke about having to reschedule their projects very often. Specifically, one of the participant mentioned that continuous follow-up on the tasks was cumbersome to keep track even with additional software support.

However, the managers at Qlik had different, but similar positions. Their style of managing personnel was very varied. While, some of the managers was organized and planned a lot beforehand, others seemed to be constantly planning when needed.

Since Qlik is working with software, the managers used some software development specific tools for tracking bugs and development. For planning, the managers used software such as Microsoft Excel and Qlik's own software Qlik sense. In addition to these softwares, three of the managers used a software called Microsoft projects, which main purpose serves as an extensive planning support tool for larger projects.

4.1.2 Brainstorming session

The sketch from the brainstorming session resulted in a mind map describing not only the visual layout of the prototype, but also functionality that needs to be included.

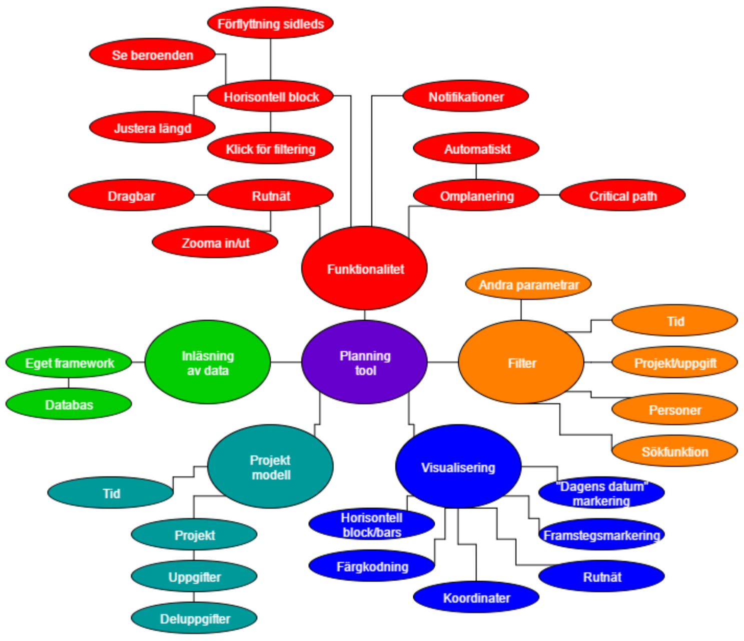


Figure 15: Result of the brainstorming session.

The result from the brainstorming session is shown in figure 15. The planning tool was divided into five categories which were functionality, filter, visualization, project model and input of data.

The functionality category contained the initial planned functionality for the extension. Although filter is a type of functionality, it is still a large enough category in the Qlik sense tool to be a standalone category in this mind map. The filter category contains all perspectives within the project. For example, one can observe the project from a time perspective or from the perspective of a software developer, where it is possible to observe the tasks of the developer and all other associations.

The visualization category contained a very rough approximation of how the extension is planned to look like. The project model contained which fundamental objects was desired in this extension. For example, a project consists of many tasks, which could consist of sub-tasks. Despite the fact that it was expressed in terms of project, tasks and subtasks it is planned to be even more general in order to fit more use cases, but the initial design will consists of projects, tasks and subtasks.

4.1.3 Lo-fi prototype

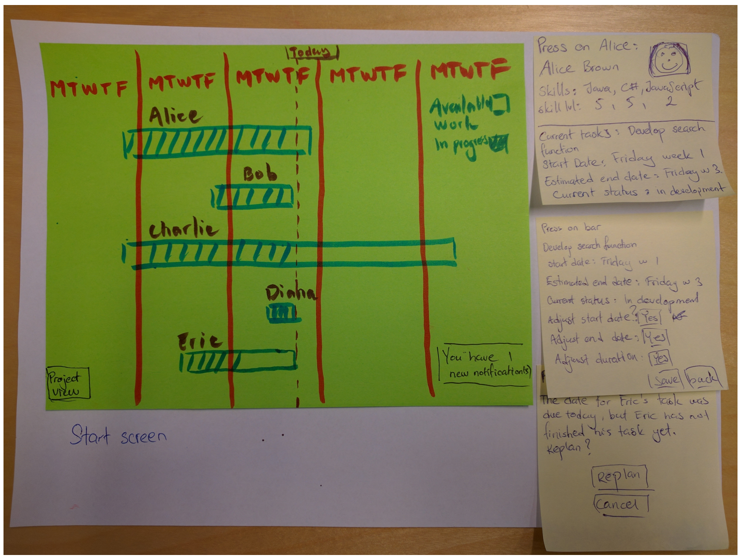
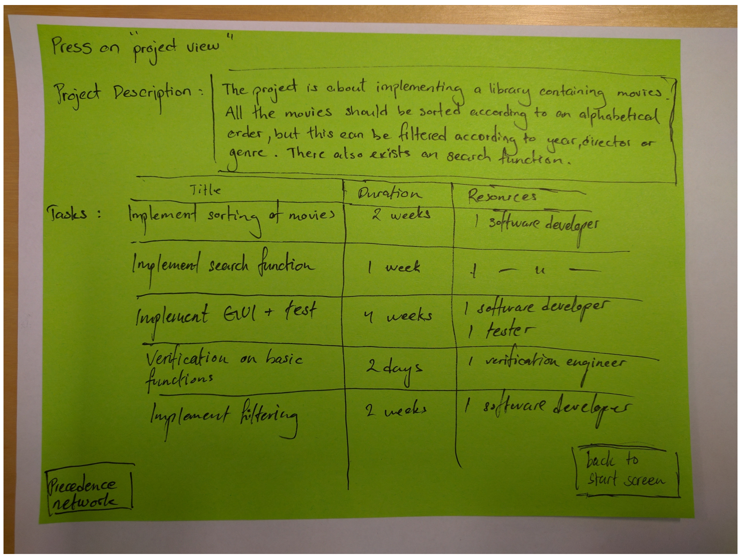

Today
MTWTF
MTWTF
MTWTF
MTWTF
MTWTF
Alice
Bob
Charlie
Diana
Eric
Availability
work
In progress
Project view
You have 1 new notification(s)
Start screen
Press on Alice:
Alice Brown
Skills: Java, C#, JavaScript
skill lvl: 5, 5, 2
Current tasks: Develop search function
Start Date: Friday week 1
Estimated end date: Friday w 3.
Current status: in development
Press on bar
Develop search function
start date: Friday w 1
Estimated end date: Friday w 3
Current status: In development
Adjust start date? Yes
Adjust end date: Yes
Adjust duration: Yes
Save
back
The date for Eric's task was due today, but Eric has not finished his task yet.
Replan?
Replan
Cancel

Figure 16: Overview of the extension of one possible lo-fi prototype in the initial phase.


Press on "project view"

Project Description: The project is about implementing a library containing movies. All the movies should be sorted according to an alphabetical order, but this can be filtered according to year, director or genre. There also exists an search function.

Tasks:

Title	Duration	Resources
Implement sorting of movies	2 weeks	1 software developer
Implement search function	1 week	1 — " —
Implement GUI + test	4 weeks	1 software developer 1 tester
Verification on basic functions	2 days	1 verification engineer
Implement filtering	2 weeks	1 software developer

Precedence network

back to start screen

Figure 17: Project view of the extension of one possible lo-fi prototype in the initial phase.

In figure 16 the first lo-fi prototype is shown. This prototype had features such as names displayed on top on the task, a dotted line to display today's date. This design was more button- and dialog based than the other. For example there was a possibility to adjust the task duration by clicking a task and then change the start date and end date via a dialog box. In addition, it is also possible to click on the name of a person located above the bar to enter the profile of a particular person. At the profile menu, one can observe the skills of a person, the skill levels and current ongoing tasks.

The bars in this extension also displayed the progress of a task. For example, it can be observed that Eric is not done with his task even though the deadline was today's date.

The lower left button switches from overview to a project view, which is shown in figure 17. This view contained project information such as a project description and a table containing the title of tasks, the duration and resources. The duration is measured in weeks but this could be adjusted according to the principles of the company. The category resources is measured in man power in this case, but could be adjusted for companies where cost of material are more significant than cost of personnel.

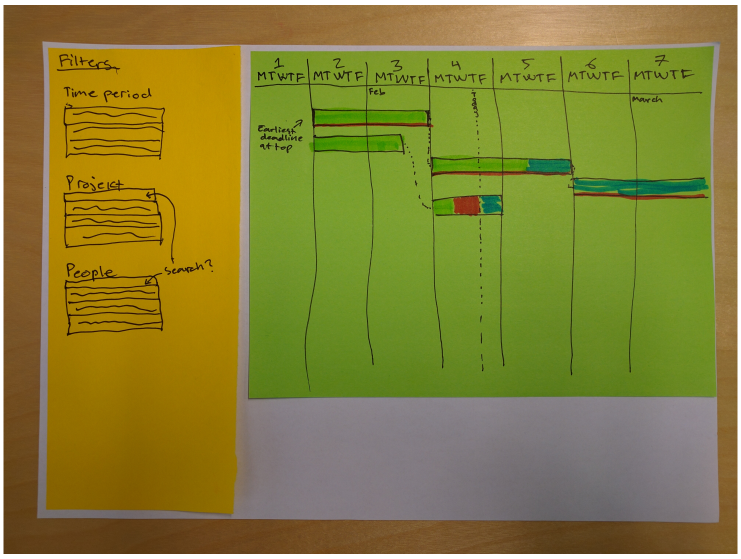


Figure 18: Another possible lo-fi prototype.

In figure 18 another design option is shown. There were some similarities between the designs, for example both had the time line displaying the current date and both had some ideas about displaying progress using the bars in some way. Apart from the similarities, this chart displays the tasks based on their start date such that the tasks starting earliest is shown at the top. This does not however had a "project view". Instead of having buttons and dialog boxes, this design focuses more on interaction with different components themselves in order to change properties such as changing the length of the projects.

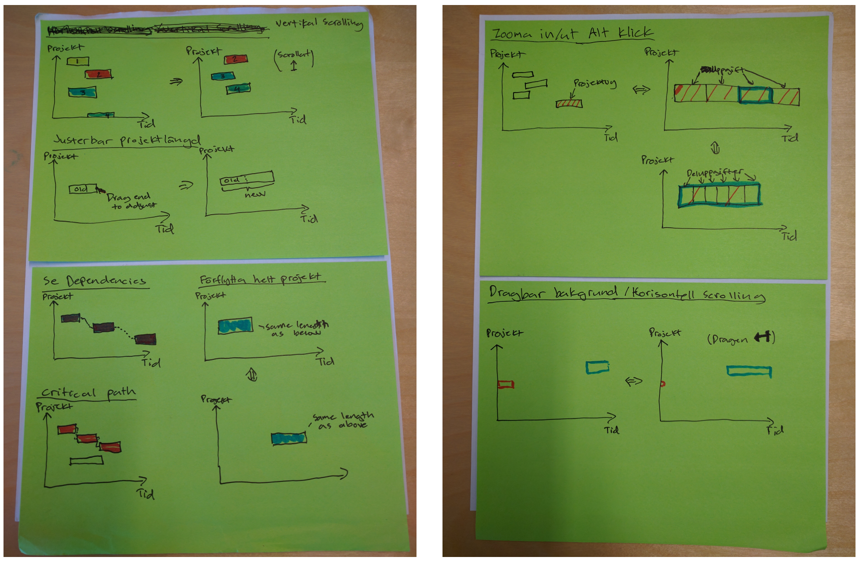


Figure 19: Interaction possibilities for the extension.

In figure 19 the different interaction ideas are shown belonging to the design in figure 18. However, these are several different possibilities of how one can interact with the extension. The figures shows interactions such as vertical scrolling, horizontal scrolling, adjustable project duration, visualized dependencies, critical path, zoom in/out, click in/out, and movable projects.

4.1.4 Hi-fi prototype

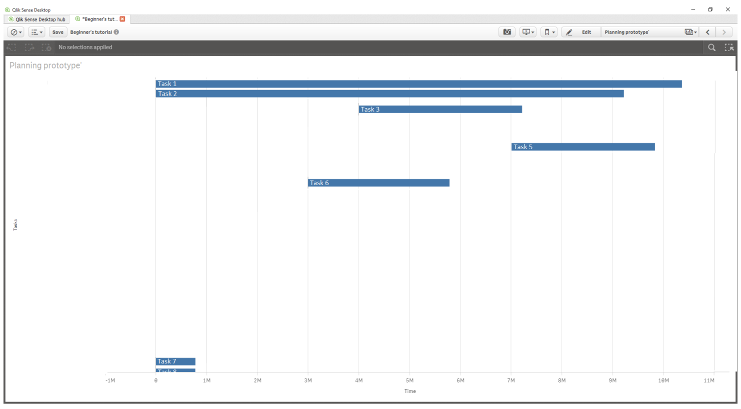


Figure 20: Start screen of the hi-fi prototype.

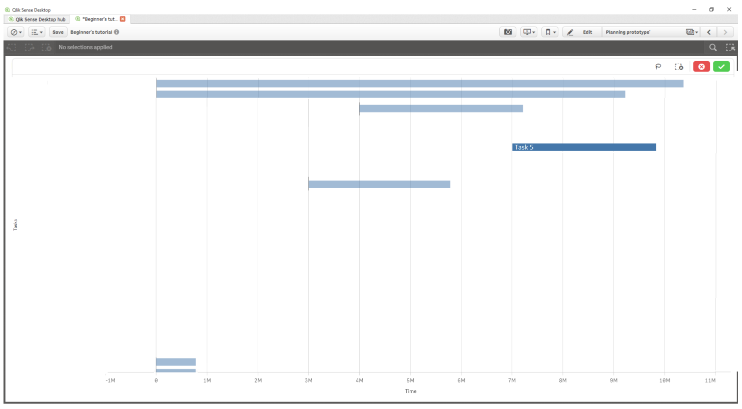


Figure 21: Screen showing one selected task in the extension.

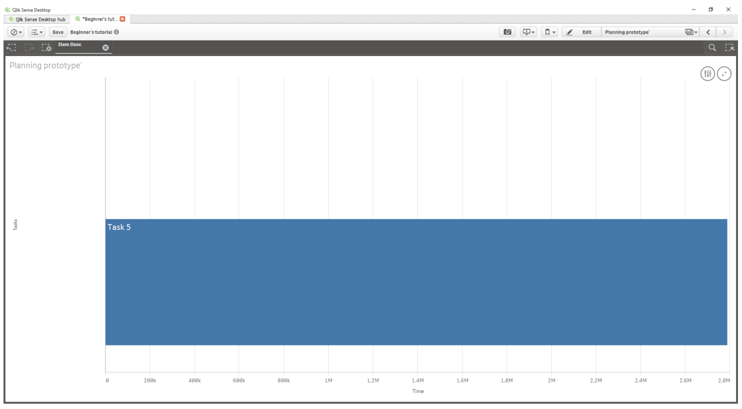


Figure 22: Screen showing one selected task in the extension.

The main screen of the extension is shown in figure 20. The hi-fi prototype was made using edited screen dumps of Qlik Sense and the prototyping tool Invision [18]. From this screen, the user could click "Task 5" in order to select it yielding a screen as seen in figure 21. The other projects are dimmed, hinting that the user has made a selection. When the user presses the green button at the top, the changes of data to display is updated and the screen shown is figure 22. If the user presses the red button, all the changes are reverted and the user comes back to the screen shown in figure 20.

4.2 Sprint 1

For sprint 1, no visual results were produced. This sprint was focused on learning Qlik Sense and D3.js for visualizing the gantt chart and the task assignment respectively.

4.3 Sprint 2

4.3.1 Task Assignment

The results from this stage, yielded the decision of continuing with the genetic algorithm instead of the heuristic algorithm.

4.3.2 Gantt Chart

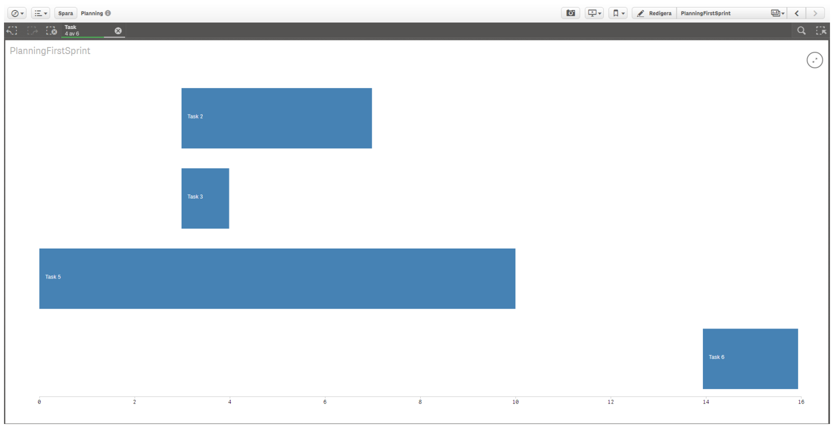


Figure 23: Four of the six tasks have been selected and the visualization filters out the non-selected tasks.

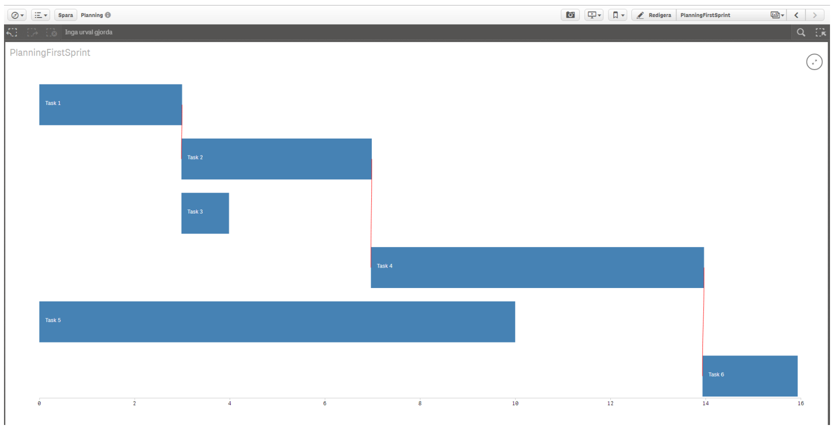


Figure 24: A time line is now visible and the numbers indicate the amount of time(measured in days) which has passed since the start of the project. The red lines shows the critical path.

Firstly, the bars representing each task are now clickable making the bars interactive. This functionality allows the user to click on bars to make selections and the extension in turn will render only the selected bars(see figure 23). Moreover, one can notice the timeline on the bottom of figure 23 acting as a reference for the bars. Finally, the red lines visible in figure 24 shows the critical path starting at task 1 and ending at task 6. This gives the user the ability to oversee the most important tasks.

4.4 Sprint 3

4.4.1 Task assignment extension

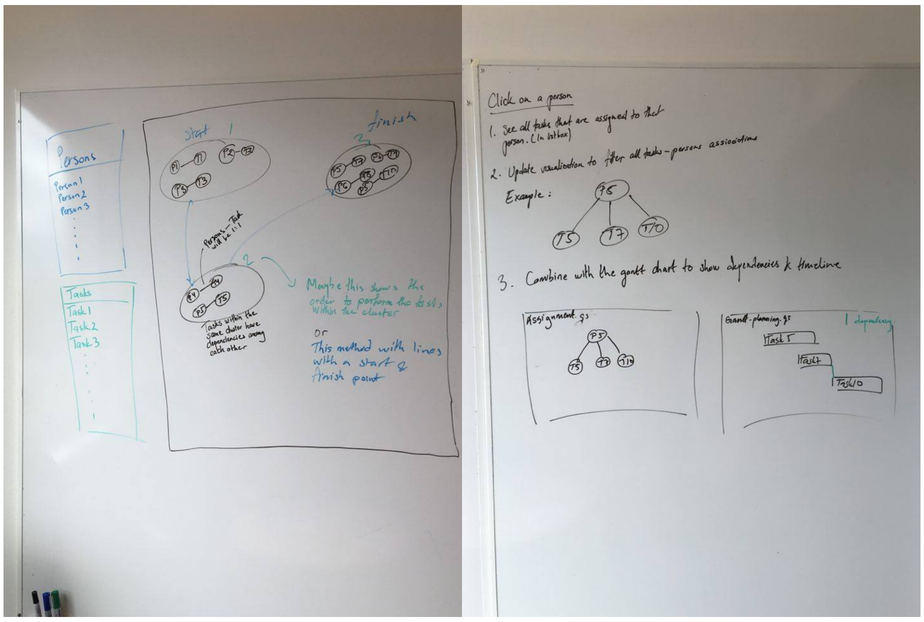

Persons
Person1
Person2
Person3
Tasks
Task1
Task2
Task3
Start
1
finish
3
2
P1
T1
P2
T2
P3
T3
P4
T4
P5
T5
P6
T7
T8
T9
T10
Persons – Task will be 1:1
Tasks within the same cluster have dependencies among each other
Maybe this shows the order to perform the tasks within the cluster
or
This method with lines with a start & finish point
Click on a person
1. See all tasks that are assigned to that person. (In listbox)
2. Update visualization to filter all tasks – persons associations
Example:
P5
T5
T7
T10
3. Combine with the gantt chart to show dependencies & timeline
Assignment.js
Gantt-planning.js
dependency
Task 5
Task7
Task10

Figure 25: Lo-fi prototype to show tasks dependencies and also task assignment.

Figure 25 shows one of the lo-fi prototypes for the task assignment problem. This lo-fi prototype contained two different extensions.

To the left the first extension is shown. It contains a lot of so called clusters, which is a collection of tasks. The second extension is a tree with one or several person root-nodes and their respective tree contains all the tasks assigned to a given person.

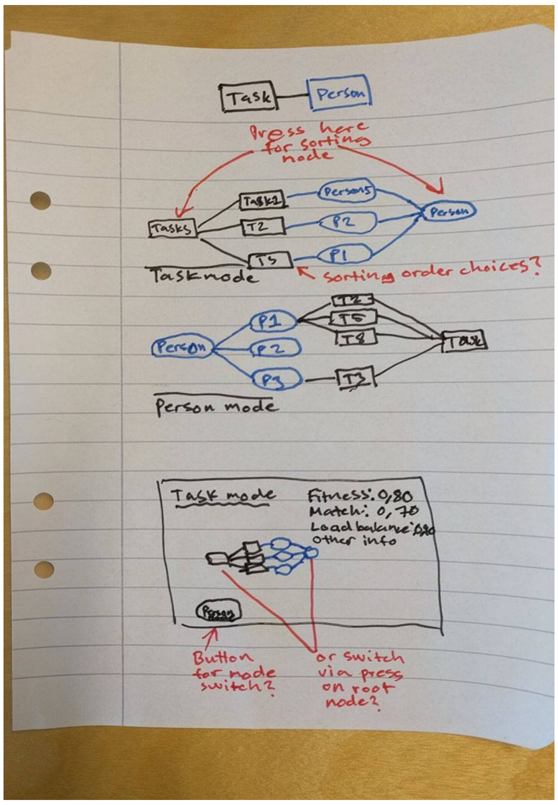


Figure 26: Lo-fi prototype for task assignment extension.

In figure 26 another lo-fi prototype to visualize the task assignment is shown. The lo-fi prototype shows a node structure with connected persons to tasks. In this extension, a user can choose from from two different modes, a task mode and a person mode. When task mode is pressed, each corresponding person to each task is shown with a connection line between the two. In person mode, all tasks assigned to one person is shown. This lo-fi prototype became the one used for developing the hi-fi prototype.

One thing which differed in the hi-fi prototype compared to the lo-fi prototype was that we had one node as root from one way, instead of one root on each side as the lo-fi prototype suggests. This was because implementation for this node structure already existed in D3js and lot of effort was saved compared to having the node structure as shown in the lo-fi prototype in figure 26.

This extension will allow the user to filter based on specific tasks or a specific person. When used together with the Gantt-chart, information such as where the tasks are planned can be seen and also which other tasks are affected when a person does not finish their task in time.

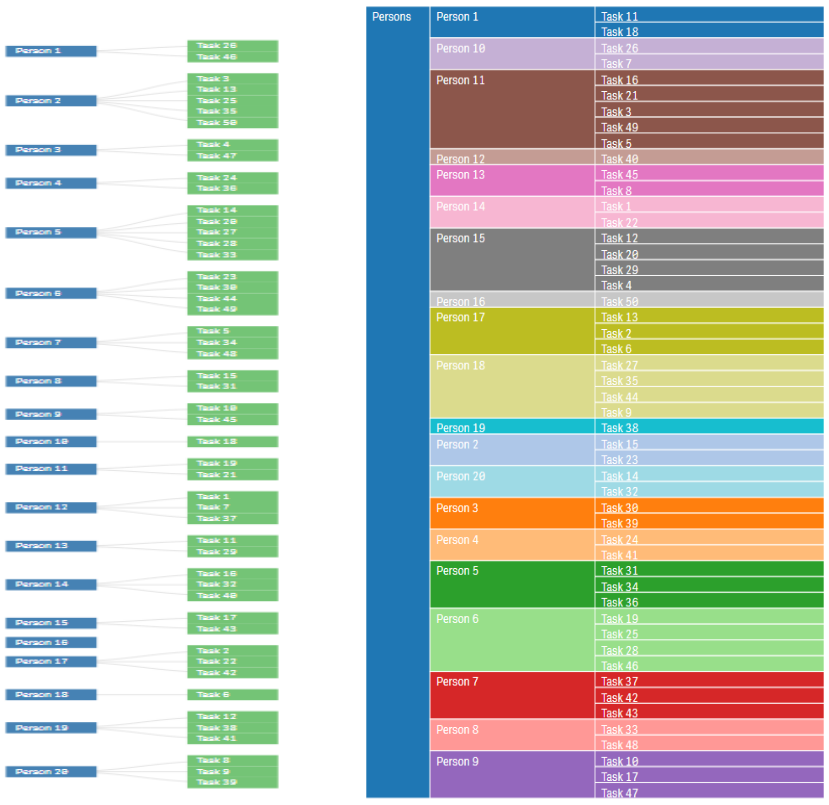


Figure 27: Hi-fi prototypes for task assignment.

In figure 27 to the left the hi-fi prototype of the above visualization is shown. The hi-fi prototype was implemented in code directly, with some of the basic functionality such as zoom and drag. When developing further on, another hi-fi prototype (right of figure 27) was created, which is based on the partition layout in D3js.

The partition layout consists of layers from left to right, similar to the other hi-fi prototype. The first layer is the root layer, which shows the title "Persons". In the middle column, all persons in the project is shown. The size of each rectangle depends on the number of task each individual has. The last layer consists of tasks which are used to show which person each tasks is assigned to. The corresponding data is color coded such that each person and all tasks assigned to the person has the same color rectangle in the visualization and likewise the other way around. The sum of the area of all assigned tasks has the same size as the person it is assigned to.

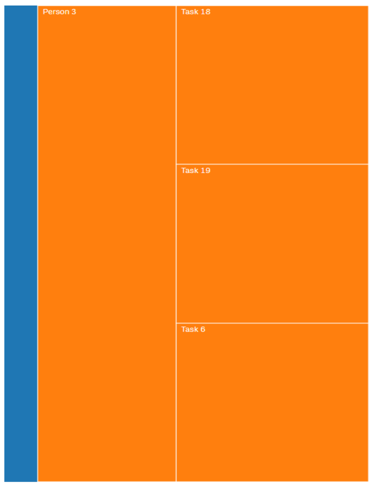


Figure 28: A zoomed in view of a person from the prototype in figure 28.

For the partition prototype a functionality to zoom in on each individual person was tested. When a person is selected by clicking, that person's assigned tasks are shown, as shown in figure 1. The user can also zoom in on individual tasks, and in a final implementation, more information could be revealed when zoomed in on a specific person.

To the leftmost in the figure, a small part of the rectangle of the layer above the current layer is shown. The idea behind leaving a part visible to the left of the outer layer is to allow the user to show that a user has in fact zoomed in and also that the user will be able to derive that that part can be clicked in order to zoom out again.

4.4.2 Gantt chart

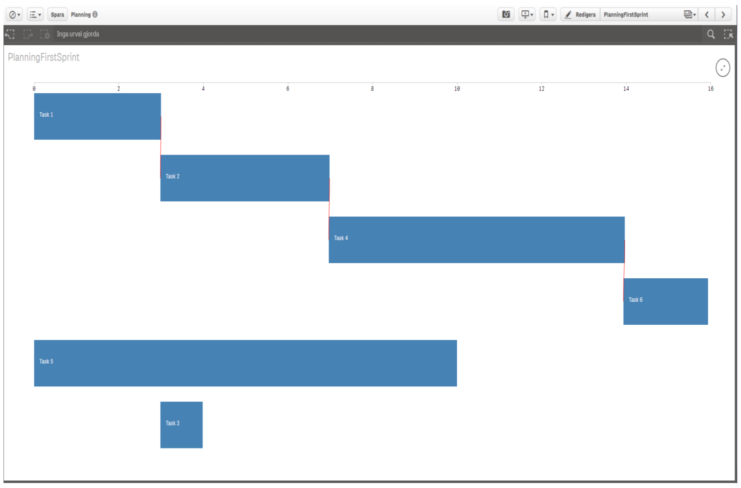


Figure 29: Shows the gantt chart before any selections is made. The red lines indicate that there is a dependency between two tasks.

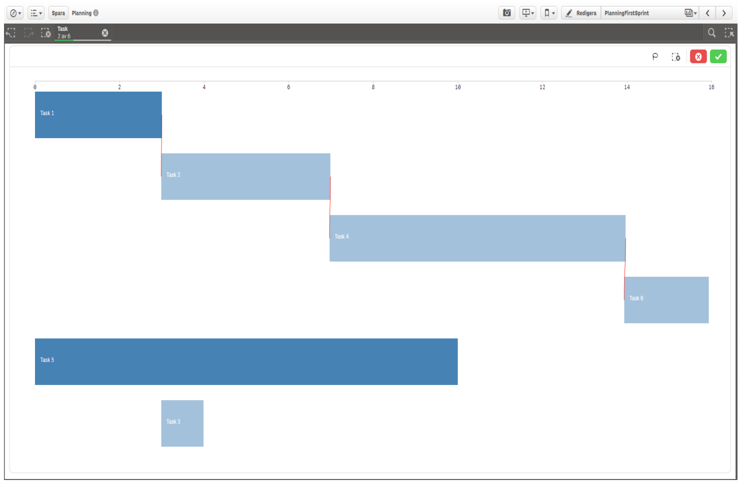


Figure 30: Displays the gantt chart in selections mode before any selection is confirmed. One can observed that task 1 and task 5 has been highlighted as possible selections.

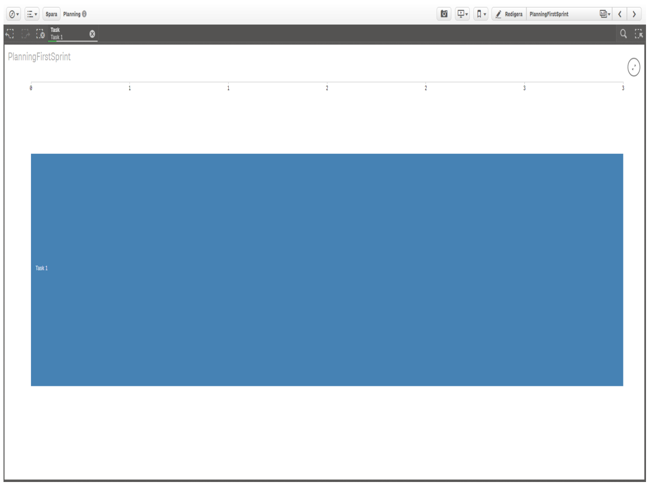
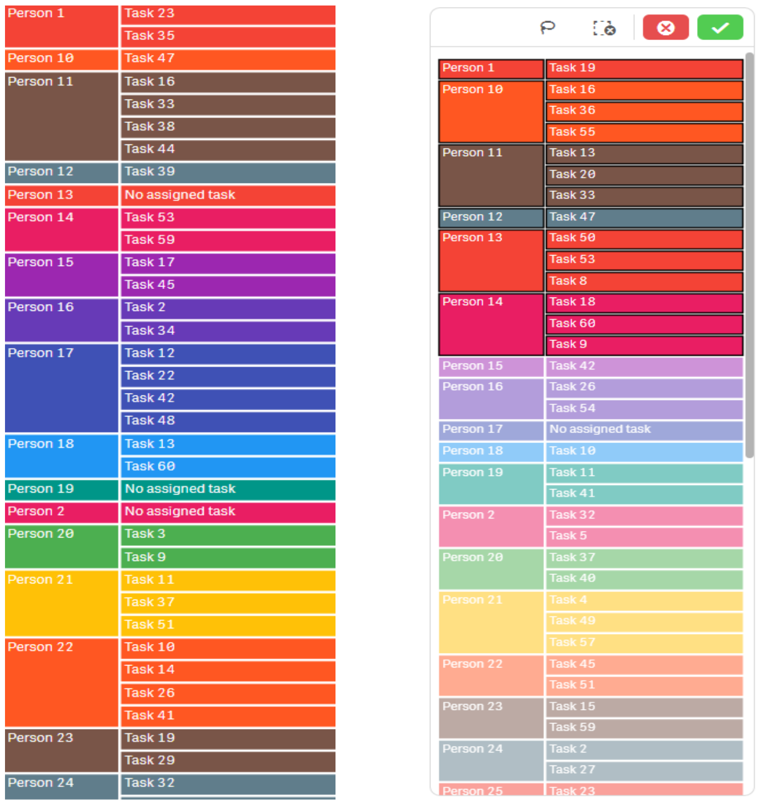


Figure 31: Displays the gantt chart for a single selection. Note that all information regarding the other bars is not available from this view.

Features added for this sprint makes it possible for the user to enter a state called selection mode before actually confirming the resulting selection. This also applies if the user is performing multiple selections. In order to confirm the selection, the user presses the green confirm button in the upper right corner. To cancel a selection, the user presses the red button in the upper right corner next to the green button(See figure 30).

4.5 Sprint 4

4.5.1 Task assignment extension



Person 1	Task 23
	Task 35
Person 10	Task 47
Person 11	Task 16
	Task 33
	Task 38
	Task 44
Person 12	Task 39
Person 13	No assigned task
Person 14	Task 53
	Task 59
Person 15	Task 17
	Task 45
Person 16	Task 2
	Task 34
Person 17	Task 12
	Task 22
	Task 42
	Task 48
Person 18	Task 13
	Task 60
Person 19	No assigned task
Person 2	No assigned task
Person 20	Task 3
	Task 9
Person 21	Task 11
	Task 37
	Task 51
Person 22	Task 10
	Task 14
	Task 26
	Task 41
Person 23	Task 19
	Task 29
Person 24	Task 32

Person 1	Task 19
Person 10	Task 16
	Task 36
	Task 55
Person 11	Task 13
	Task 20
	Task 33
Person 12	Task 47
Person 13	Task 50
	Task 53
	Task 8
Person 14	Task 18
	Task 60
	Task 9
Person 15	Task 42
Person 16	Task 26
	Task 54
Person 17	No assigned task
Person 18	Task 10
Person 19	Task 11
	Task 41
Person 2	Task 32
	Task 5
Person 20	Task 37
	Task 40
Person 21	Task 4
	Task 49
	Task 57
Person 22	Task 45
	Task 51
Person 23	Task 15
	Task 59
Person 24	Task 2
	Task 27
Person 25	Task 23

Figure 32: Hi-fi prototypes for task assignment.

In order to differentiate between selected and non selected data there was two different effects included. When a new selection had been made, the user got direct feedback by having the selected values get black edges around it and all corresponding data.

When the left column (the person column) was pressed, all the assigned tasks was selected as well and likewise when one task was pressed, the person assigned to it and all sibling tasks got selected as well.

To make the distinction from the selected values and the non selected values, the non selected data did not have a black edge and also the opacity of the non selected persons was lowered, making the selected task more visible compared to the non selected ones.

There are two ways to select the persons and corresponding tasks in this prototype. The first way is to click on one person in order to select the person. The second way is to hold down the mouse and move it above multiple persons/tasks in order to select the data. This is the way it works in many extensions found in Qlik sense and it was therefore built the same way in order to aid the mental model of an experienced Qlik sense user.

In the previous version of the prototype, the visualization was scaled to fit all information onto the screen. This means that all persons and tasks was always visible, no matter how many persons or tasks that was available. This becomes a problem when the given project has too many persons involved or many tasks. Of course, one could set a limit on the maximum allowed persons in a project or use the ”zoom in on a person”-function. But since the ”zoom in on a person”-function was removed there was just one reasonable workaround, having a maximum allowed number of persons in a project.

Realistically though, the number of persons depends from company to company. In order to better reflect the reality, a scrolling bar was used instead, giving rectangles representing tasks a smallest possible height. This way many more tasks could be shown without having to zoom in on any particular part of the extension.

4.5.2 Gantt chart

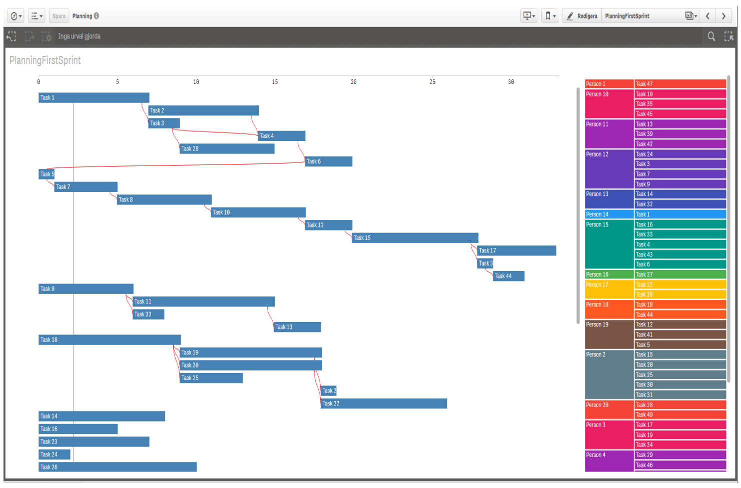


Figure 33: The integrated Gantt chart with the task assignment.

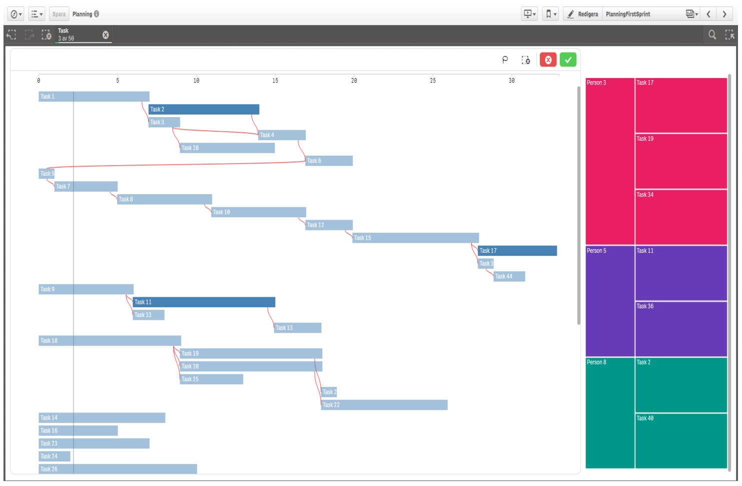


Figure 34: The integrated system in selections mode. Notice how the extension on the right hand side filters out the number of persons and tasks. As before, the extension on the left retains its opacity for selected tasks.

In figure 33, the user can observed the default state when accessing the app and it shows the two extensions side by side.

In this state, the user is able to scroll among all tasks, compare the current date relative to the end date and observes the interaction between the extensions as users makes selections(see figure 34). In addition, all bars are now sorted in an order of importance and priority.

4.6 Sprint 5

4.6.1 Task assignment extension

The results of sprint 5 for the task assignment extension provided no additional visual features as mentioned in the method section, but the algorithm was improved, resulting in a better fitness score.

4.6.2 Gantt chart

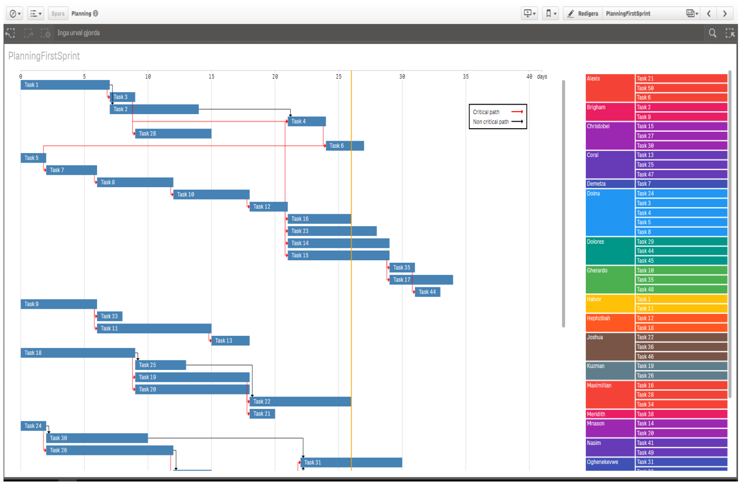


Figure 35: The integrated Gantt chart with the task assignment for sprint 5.

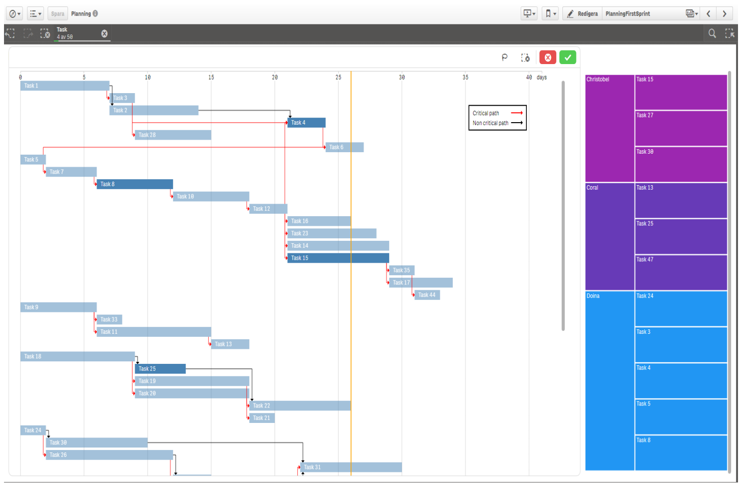


Figure 36: The integrated Gantt chart with the task assignment in selection mode.

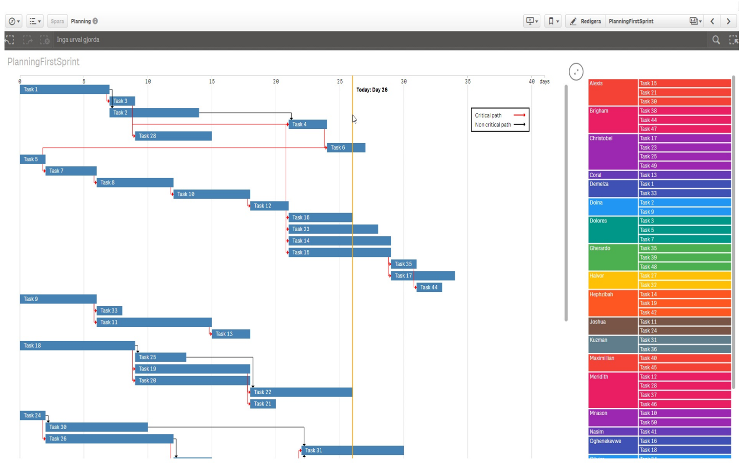


Figure 37: While hovering on the orange vertical line, it is possible to observe the current day of the development process relative to the timeline at the top of the canvas

All changes in this sprint can be observed by comparing figure 37 with figure 33. The dependencies are separated into critical and non-critical path, the orange line marking the date of today and finally the on hover function which will prompt the date of today in text form.

4.7 Usability testing

For the results, all participants detected the orange line indicating today's date and by making that discovery, everyone could distinguish between completed tasks, tasks in progress and task that have not yet been started. However, two out of four participants commented on the fact that the critical path was not correct and was slightly confused that it existed that many critical paths for the project. In addition, these two participants suggested that all the critical dependencies should be located in a list and it should be possible to select a chain from the list, instead of manually identifying each individual task and connect the dependencies via numerous mouse clicks.

There was one feedback that was unanimous for all test users, it was difficult to read out the duration and the start- and end date in the Gantt chart extension. Another flaw that was pointed out by three out of the four test users in total was that the legend was stuck at the top and likewise the today text shown on hover on the today time line, meaning that when the Gantt chart extension was scrolled down, the legend was no longer visible since it was stuck at the top of the page. One test user had difficulty spotting the legend due to the white background around it. The task assignment extension had positive feedback and the users felt that it was very clear which person that was assigned to which tasks. The users also had no difficulty seeing the interaction between the two extensions. when the test users interacted with one of the extensions, the users saw that something changed in the other extension as well.

For the second round of usability testing, the improvements can be noticed as all of the participants can easily remark on when tasks are supposed to be started and when tasks finishes. However, all of them also noticed that when hovering over a bar, no tooltip was displayed as this is the standard in Qlik sense, it was expected to occur. This was the primary remark as all the participants wanted the tool tip feature. Two of the participants also commented that the prototype needs more of an overview and one of them even commented that it would be suitable to integrate the task assignment extension into the Gantt chart extension into one single component.

4.8 Algorithm

A genetic algorithm was created for solving the task assignment problem. The genetic algorithm was written in pure JavaScript. A genetic algorithm tries to mimic the natural evolution process. It mimics the evolution process by creating a population of solutions, and then generating new solutions as combinations of the old solutions. Creating this algorithm was the most time-consuming part of the project and took more time than creating the visualization.

How good a solution is depends on their fitness score, which is a measurement on how well the solution fulfills the criteria of the given problem. The fitness score for the task assignment problem is determined as the average of parameters used. All values are in range from zero to one. If the fitness score is one, then the optimal solution has been found and all the the requirements of the parameters have been fulfilled. There are two parameters to be set for the algorithm. First is the match rate, which shows how well a person is suitable for the assigned task and the other is workload, which takes into consideration of how much the person is already working on. In the algorithm, there are settings to alter the weight of the parameters. For example if the weight for match rate is two and the weight for workload is one, then the algorithm will take more into consideration of the match rate than workload. The resulting configuration will have qualified personnel on tasks, but they might have to work longer hours.

The genetic algorithm iterates until the fitness score of the best solution in the population stagnates and does not improve or until it has run a fixed number of generations. Initially, a population of random configurations of persons to task were used as a input to the algorithm. These configurations were then combined and created new configurations. The combination step is run until one has a new population at the size of the old population. At this stage one has two population of solutions, an older population with the old solutions and a younger population with the new solutions created by combining the old solutions. These two populations are then combined to create a final population for a generation. The new generation is created by replacing the worst performing solutions in the older generation by the best performing solutions in the younger population. How many to replace depends on the ratio chosen. In our algorithm the ratio 50% is chosen. This ratio was chosen to not completely dismiss any good solutions from the older generation and at the same time not add too many new solutions which might be worse than the older generation's solution.

If the user have the required domain knowledge, it is possible to set advanced settings for the algorithm, such as mutation rate, number of iterations, population size and so on.

The result from the genetic algorithm varied from each time it was run, but for a sample set of data consisting of 20 people and 50 tasks, the fitness score was usually around 0.8. The settings was altered to the most favorable condition as possible.

5 Findings and Discussion

5.1 The design process

The design process used in creating the planning tool was combined with the software development process and added as deemed necessary. Some sprints did not have any design process at all since there was a lot to develop which did not require a graphical user interface. For example, in some sprints most work was developing and finding the correct algorithm to use and did not have any connection to the design process.

The design process used in this project consisted of interviews, brainstorming, lo-fi prototyping, hi-fi prototyping and a user study. In this section, the design process and what changes we would make is discussed.

5.1.1 Interviews

The primary reason to conduct interviews instead of surveys was to explore the whole chain of events and all associated interactions when it comes to planning. In this case, it is important to evaluate the user experience from the decision of management to undertake the project to the completion of the project. Every step of the management process is crucial for the designing prototype and a good approach is to let the participants answer the questions freely in order for us to gain clarity regarding the subject.

If surveys were to be conducted, then the requirements imposed on us will be quite high. Specifically, we need to have previous knowledge of software project planning, in order to ask useful questions and utilize the answers to create a lo-fi prototype. In addition, the context could be lost in a survey, whereas in a interview, the flow is kept continuous(as the participant is allowed to speak freely) making it easier to analyze the many situations that managers have to tackle every day.

Furthermore, surveys are also less flexible, which means that every question needs to be planned ahead and there is no possibilities given to readjust the questions once the surveys are handed out. However, this is not the case of the interviews and the objective of the interview can switch to focus more in depth on the opinions of the participant, making interviews very dynamic [20].

Although, interviews will often yield the whole user experience, it takes a lot of time to conduct interviews, while it is much quicker to produce a survey and send it out via various medium such as social media, web forums, university web pages and so on. The intended audience of the survey was project managers and the authors felt it was easier to conduct the interviews on site as well.

In our case, the results of the interviews was as expected, we got feedback which represented their thoughts from the start of the planning phase throughout the entire project. It also made it easier for us to step into the minds of a project manager, which simplified the design of a prototype from their perspective. However,the insights also made us understand the complexity of planning and even with the assistance of software, planning was not bound to be easier. At that point, we understood that the visualizations created had to be intuitive and pre-cognitive in order for it to be usable.

5.1.2 Brainstorming

The brainstorming session held at the early days of the project was good to get the project started. It also provided various ideas for the prototype and eventually these ideas mostly resulted in features that could be implemented. This was beneficial in the sense that the brainstorming session could be used as product backlog and one could assign a priority to each feature based on the difficulty and necessity of the feature.

On the other hand, brainstorming sessions comes with backlashes as well such as the scope of the project was skewed due to a humongous amount of ideas/features being generated during the session. Therefore, it was not feasible to implement all the ideas and some in our case, were discarded immediately for various reasons. For example, one of the ideas was to implement a standalone framework and database to be used as reading inputs/writing outputs, but that was too cumbersome to implement and in addition, Qlik sense already have a database and reading inputs proved to be relatively smooth as well. In conclusion, the masters thesis have a clearly defined scope, when referring to the allocated time and therefore one must implement the core features first before working with excessively subtle visualization details.

5.1.3 Lo-fi and hi-fi prototypes

For the Gantt chart extension, two lo-fi prototypes and one hi-fi prototype was created. For the task assignment extension, two lo-fi prototypes and two hi-fi prototype was created. For the Gantt chart extension, it felt like the Gantt chart was already a well established chart. For the task assignment extension however, we had no idea how to visualize it since there are not much guidelines for it.

In retrospect, it feels like we could have explored more design options before going to the next step of the prototyping process. But due to the time restriction, it was difficult to estimate how much time we could spend on the prototypes before deciding for a final prototype version. It was also unclear how long the algorithms would take to implement which put a limitation on how much time we could take. Furthermore, our goal was to produce a prototype, where proper interaction is taking place, which will make the user think that the product is almost done. We felt that this effect could not be achieved by showing a hi-fi prototype as the end result.

There was also another drawback due to the time limitation, namely that there was no user study or focus group to test the lo-fi and hi-fi prototypes. Instead, we had to go on our intuition and discussion within our group in order to decide how the prototypes would look. Because of this it is difficult to verify if the right decisions were made in the early stages.

5.1.4 User study

Although, a formative approach was chosen as the study model, there were only two iterations involved in the process. While these iterations provided the necessary feedback required to identify design flaws in the prototype, the study can not prove how good the usability is. In other words, there is no metric scale for usability as one can not simply say that the usability score is at a specific percentage.

Furthermore, the number of participants was capped at 10 and the subjective opinions of the participants might not apply for the general population. For example, there was a consensus among the participants that there was a connection between the tasks and persons, but there are no statistics to support this claim [5]. In our case, it is not appropriate to calculate confidence intervals and SUS scores, because of the low number of participants and that the study is formative. However, there is always room for improvement and it has been agreed upon that, given more iterations, the design of the prototype could be more powerful due to more participants which will yield more feedback.

Even if the amount of participants are low, it was still possible to draw some consensus for the visualizations, as can be seen between the two fixes, where additional gray lines were added to in order to make it easier to identify when tasks started and ended. This effect was also prominent during the second phase of the usability test as all the participants were able to correctly tell when the tasks started and ended.

For our user study, the users provided a lot of constructive criticism, which made it easier for us to identify the flaws of our prototype and apply quick fixes to these problems. This proves that only by selecting a small set of users for the study, we were able to identify some problems at an early stage. While either one of us advocate for our custom approach, we believe that the combination of a formative approach and a summative approach would yield better results as it could verify that the design holds for a larger set of users. In the end, if you are short on time, as we were, it is sufficient to run the user study on a small set of users.

5.2 Design decisions regarding the prototype

5.2.1 Lo-fi and hi-fi prototypes for the task assignment extension

Lo-fi and hi-fi prototypes were created after demand. The prototypes gave us a good initial design on the extensions. Two lo-fi prototypes were created for the task assignment extension.

From one of the lo-fi prototypes (figure 25) we got the idea to use clusters, meaning that one dependency chain of tasks could be put together to a so called cluster. The idea behind the extension to the left in figure 25 is that tasks are arranged after the tasks dependencies into so called clusters. For each task in a cluster a person node is bound. The reasoning behind this idea is that people working in the same cluster must communicate with each other since their tasks depends on each other. Therefore this extension would show an overview of all clusters and when a person is chosen(filtered) all the corresponding clusters to that person is shown. The clusters were not applicable for the task assignment extension, but more for the Gantt chart, but the idea arose from the lo-fi session for the task assignment.

The other lo-fi prototype (figure 26) showed a node structure, and this was the idea which we decided to build upon. We decided to use a horizontal node structure instead of a vertical node structure, because the mapping of the text would be more intuitive this way, since most text is read horizontally and not vertically.

For the task assignment extension, two hi-fi prototypes was also created. The hi-fi prototypes (figure 27) were implemented in code using D3JS directly. It was implemented in code because the structure was easily created in D3JS and it was our opinion that it would take more time to create a visualization using a mock up tool. Furthermore, if the design was already built in code, an evolutionary prototyping approach could be used where we built upon the already existing design.

Analyzing the hi-fi prototype shown to the left in figure 27, a realization came to mind. There was too much space which went unused, and for large projects it was difficult to get a good overview. Another realization was that it was enough to have the person view, instead of both tasks and person view, since the Gantt-chart works as task view. The idea was that when one or multiple tasks are selected in the Gantt-chart, the corresponding persons are highlighted in the task assignment chart as well.

In this step it felt like we would have to return back to the drawing boards before creating a new extension. However, looking through D3JS documentation we found the partition layout, which our final prototype is based on.

Unfortunately, there was a problem with the first partition layout prototype. Having the ”zoom in on person”-feature did not allow for another feature, namely the feature of filtering. Most Qlik Sense graphs allows a user to select a subset of the total data in order to only retrieve the matching data for the selected data. Filtering in the task assignment extension is the ability to select one or several specific persons to get the corresponding data to the selected persons. Therefore, it was decided to remove the zoom feature and instead replace it with functionality for selection similar to the way other Qlik sense extensions work.

5.2.2 Lo-fi and hi-fi prototypes for the Gantt chart

Referring back to figure 18 we realized that not all above mentioned features could coexist together. For example either vertical scrolling and horizontal scrolling could be used to move around in the extension, or the scrolling wheel could be used for zoom in and zoom out. If zoom in and out is used, then we could instead add a ”drag the background”-feature in order to move around the chart. The different tasks should be selectable, which means that when a user clicks on one of the tasks, only data corresponding to that tasks should be shown.

At the infancy of the design process, two lo-fi prototypes were created from the feedback of the interviews and brainstorming session. Due to the amount of feedback, we decided to create two horizontal lo-fi prototypes in order to explore all the visual and interaction options before deciding on a prototype to be based on. Furthermore, horizontal prototyping suits Qlik Sense better as a single interaction can affect the whole application. It is therefore imperative that we understand all events occurring at a high level and their implications instead of examining small isolated events.

As for the lo-fi prototypes, the first one(see figure 16) looks somewhat similar to the final prototype, but the interaction is completely different. As mentioned above, the design of the first lo-fi prototype is more of a button, dialog and radio button based approach. This design felt more outdated and obsolete, which prevented us from implementing it. Furthermore, it was regarded as being too complicated and non-intuitive for the user to utilize it effectively. In addition, it seems that adding so many buttons will just overwhelm the user and will ultimately make the prototype less usable. Adding more features, does not necessarily mean that the resulting product will be more usable, on the contrary, it might prove better to keep the design as simple as possible. For example, in the first prototype, there are two ways to get the duration for a particular task. One can either press the project view button on the lower left of figure 16 or press a bar. A better solution will be to find one universal way to find the duration in order to not confuse the user [9].

The second lo-fi prototype(see figure 18) was more similar to Qlik sense, in terms of interactability. When selecting items in the second lo-fi prototype, the visualization filters out the dissociated values, similar to the actions in Qlik sense. Let us not forget that the main core of interaction of Qlik sense is the green, white and gray association, where green values are selected values, white values are values that are associated with the green values and finally, the gray values are the dissociated values. Our objective was to maintain the natural interaction model in order to address the experienced Qlik sense user. In that way, the experienced Qlik sense user will already be familiar with our app from the start and they will have no problem interacting with it, let alone use it. However, for the new users, we believe that Qlik sense is relatively easy to learn and new users to Qlik sense will have a learning curve in the beginning, although the process will be fast and relatively smooth if one follows the tutorials.

As the second lo-fi prototype was chosen, the resulting hi-fi prototype was based on that one. It can be argued of why we did not create several hi-fi prototypes and merge the results together, as this could have provided us with more optional prototypes to explore the interaction. However, at this point in development, we have no idea how the user is going to interact with the prototype. Therefore, it was decided, rather quickly that we need to design a hi-fi prototype, implement it and design a usability study, in order to gain some knowledge regarding the end user. Then, we will re-iterate the process, which will hopefully provide small improvements between the iterations.

In our opinion, the result from the interviews provided a lot of different opinions regarding planning and for the hi-fi prototype, we thought it will be unwise to implement all the requested features, just to address everyone's needs. Therefore, the hi-fi prototype was designed vertically first to include only the necessary features such as, displaying tasks relative to the timeline, selection of tasks, confirm selections and cancel selection. By using these basic features, we try to early on, include the concepts in interaction design such as feedback, when the user clicks on a bar, then the selected bar is highlighted and the rest are semi-transparent. The goal of this action is to give the user feedback that an action has been taken [9]. Therefore, when the user presses a bar, the application enters a selection state, where the user is presented with a green confirm button, a red cancel button and the highlighted bars and semi-transparent bars. Here, it is important to indicate to the user that a selection has not been made yet, until the user presses the green confirm button. When the user presses the green confirm button, the visualization will filter out the non-selected tasks in order to tell the user that the selections made has been successful.

Feed forward is also included by presenting the user, several actions from each state except from the first state, where the user must discover by themselves if the bar is clickable or not. To indicate that something is clickable, one could display a label over the bars, when hovering over one with the mouse pointer. However, we agreed that the hi-fi prototype should be kept as simple as possible for now.

5.2.3 Task assignment extension

In the beginning, the thought was to add the assigned person to the task in the Gantt chart. However, that design would lead to a hard overview of other tasks performed by the same person. Therefore it was decided to create the task assignment as its own extension.

In the early stage of the task assignment extension, there was no interaction with the extension. The only focus was to find the appropriate visualization. The first visualization was a node structure. However, we soon realized that there was a lot of screen estate not used by having the node structure. Therefore, a partition layout was used instead.

At the starting screen of the extension, we wanted to keep the number of details to the bare minimum, in order to not overload the user's attention. This was important because we had two different extensions with different information, and if too much information was displayed at once, the extension would be more difficult to understand. We decided to only display the person and all assigned tasks in the extension. For the person, other possible data to display could have been match rate, workload, load balance score. For tasks, a longer description or required skill set could have been shown.

For the start screen, it was important that the user could easily see which tasks that belonged to which person. Therefore, colors were used to show the correspondence between the person and the assigned tasks. There was one potential problem with using the colors, which was that the colors could perhaps be interpreted differently. For example, one could think that the red color would mean that there was some danger with the assignment. However, after conducting the user tests it was concluded that none of the participants, whether they were expert users or beginner users, had any difficulty understanding what the colors were for.

Furthermore, the person block was designed to take up as much space as the number of tasks assigned to the person. If no tasks were assigned, a block with the text ”No assigned tasks for person x” was shown. If it was not designed like this, then it would be difficult to tell which person a specific task belonged to, since it could have belonged to the person above or below it.

After having the static look of the extension, we wanted to create affordance for the extension, allowing the user to understand that one could interact with the task assignment extension. Therefore, a hover function was added. When the user hovered (held the mouse pointer) above a rectangle, the person and the corresponding tasks got a black edge around them, indicating what the choice would be. When the user stopped hovering above one set of data, the rectangle edge once again became invisible. It was important to highlight all belonging data. If only the person would have been highlighted when hovered on, and tasks highlighted when hovered on, the user could think that there are two separate selections to be made in the extension, which is not possible.

When the user had selected one or several values, we wanted to show which selections that had been made in an easy way. Therefore, feedback was added in the form of highlighting the selected values by having max opacity and half opacity for the non selected values. This makes the selected values appear in a bright color while the non selected values was shown in very light color.

We wanted to make it clear that when one extension was interacted with the other extension also updated its data. A transition was added for this purpose. For example, when a task was selected in the Gantt chart, the task assignment extension changes the information to only display the corresponding data for the selected task.

5.2.4 Gantt-chart

As mentioned above, the gantt-chart was implemented in five sprints, each lasting two weeks. This does not mean that the design was neglected during these sprints, in fact many major design decisions was made during this phase.

At first, we wanted to implement the hi-fi prototype in order to get some interactions in the prototype. However, it was decided that no user study should be conducted yet as even though the user could interact with the prototype and get feedback, the results were misleading. In other words, the user would not be able to interpret the results and there are no goals for the user to achieve during the end of this iteration. To clarify, the user had the ability to filter tasks, but as that process is performed the selected tasks were displayed and the bars got bigger. This provided no data discovery as well and most Gantt chart does not support this functionality as their intended use is to observe the whole project from the start of the project to the end of the project. Even for us at this stage, it was unclear what we wanted to achieve with filtering on the Gantt chart, as we have never seen a Gantt chart with a filtering functionality before.

The rest of the prototype are simply small visualizations needed either to support the data discovery for the user or to highlight important properties that will be utilized by the user. For example, the red lines highlighting the critical paths were chosen to focus the user's attention in regards to the color red implicating some danger. This forces the user to explore the relationship between the tasks in the critical path. In contrast, the non-critical paths are black, because the priority of these tasks are lower than tasks within the critical paths and the attention to those task could be lower.

In addition, the timeline was added, because the user needs the line as a reference point to figure out the duration of a task. Without, it will be near impossible to observe the actual makespan of a task and it will only be possible to compare the tasks relative to the length of the bars. By adding a timeline, it is justified to also add a line indicating the date of today, in order to highlight progress on tasks. One can discover, tasks that were supposed to be done by today, tasks currently in progress and finally tasks that have not yet been started.

In Qlik sense, the affordance is achieved by hovering over an object and a label displays, showing various information regarding the object, as mentioned in section 4.6.2. This invites the user to click on the object, whether it is a bar or a point. In our opinion, both methods for affordance is equally good, but task assignment extension is thought to be working as a label displaying the name of person, the description of the task and the duration. Therefore, we opted to skip the natural on hover function in Qlik sense, because in our case, the affordance needs to be clearer as the gantt chart is almost static.

5.3 Future work

In retrospect it seems that we had the wrong idea of how the Qlik sense framework was supposed to work. In the initial session, there was a lot of features where one could change the data of the task such as changing the duration and start- and end dates of the tasks. This was before realizing that Qlik sense was more used to visualize data and not for modifying the underlying data.

The current version of the planning support tool should not be viewed as a final version, but more of a early prototype, which one could build upon. There is several improvements to be made for this tool. Firstly, the current tool solves a very specific problem with very specific parameters of what defines a task and what a resource is. This of course varies from one industry to the next.

Having access to the extension code, a company could optimize it in order to make it work for their structure. In software projects, a resource could be a person, which have one mode of executing a tasks, i.e. to implement the software. But in a sewing company a resource could for example be a machine, which could have several modes, such as running it automatically or manually. Depending on which mode is run, the tasks could have different duration, which is something that the company would have to handle in the planning tool.

The most critical improvement is to change the structure of the program. In the current version of the tool, there is no way to save a given task assignment. This must of course be handled before releasing the tool. In the current version, the task assignment extension runs the genetic algorithm once and then uses the results from this task assignment in order to show the connections. This means that the person to task assignment is given in software, as opposed to the more natural Qlik sense method, which is to load the data from a file.

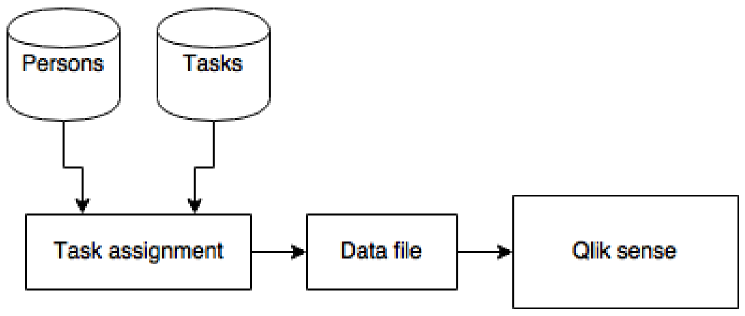


Figure 38: A proposed structure of the tool for future improvements.

Our proposal, shown in figure 38, is that the software is divided into two parts. The first part is responsible for generating a task assignment. It has all data stored in databases that is necessary to calculate the task assignment and all data regarding positioning of the tasks such as earliest and latest start and end dates. This program could show what the genetic algorithm score would be and also have options to select which persons and tasks that are active in the given run of the algorithm.

When all information has been calculated, this data could be stored in some data format such as an Excel sheet or a CSV-file (comma separated values file). Data is then loaded into Qlik sense in order to visualize the connections between the persons and the tasks.

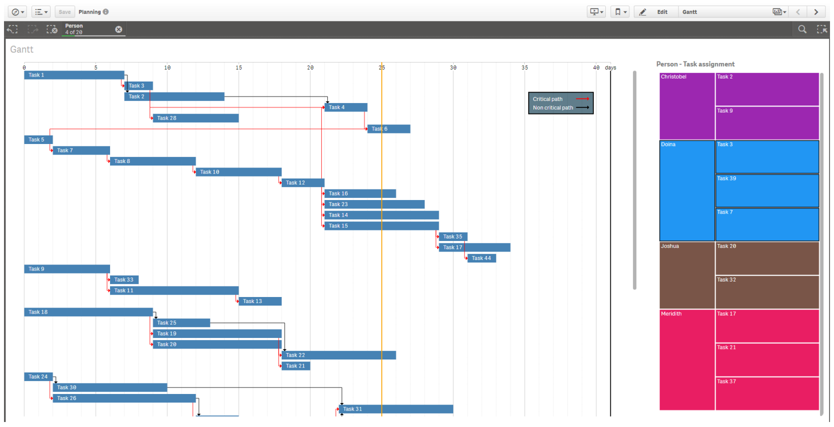


Figure 39: A proposed structure of the tool for future improvements.

This fixes one existing bug in the Gantt chart extension. There is currently no association between a person and the tasks assigned to the person loaded into Qlik sense. The association only exists in software since the genetic algorithm creating the task assignment runs in the extension, and therefore when a user filters the extension based on person, no data will be updated in Gantt chart. This can be seen in figure 39, where four persons are selected but no tasks has been highlighted in the Gantt chart.

In the current version, when tasks are selected in the Gantt chart, the task assignment extension automatically updates to add and show the person the selected tasks belong to and also all other tasks the same person performs. When a person or task is selected in the task assignment extension, the extension performs a selection on both the persons and the assigned tasks. However, the Gantt chart only reacts on selected tasks.

However, a division of the program into the two parts is time consuming, since one could not utilize the loading of data which exists in Qlik sense. One would have to write all database queries to get the required data. The advantage of dividing it into these part, is that the algorithm part could then be written in a faster programming language than JavaScript such as in C++ in order to optimize the time consumption of the algorithm and the user could still use the results in Qlik sense.

Another advantage is that one could create a database of all employees at a company with their skill sets and then use the new program to load the persons into their current software project. If the current structure was to be used, then each software project would have to write the same person multiple times into different Excel sheets or some other data file. But using the database structure one could instead reuse a user data when the user is about to start in a new software project.

For the visualization part of the new tool, one could add even more information. In the current version, only person to task is shown in the task assignment extension and the tasks are shown in the Gantt-chart extension. One could add more information such as the match rate between the task and the assigned persons. One could also add a more elaborate task description in both extension. From the interviews conducted in the start of this project, one of the managers said that it would be good if the extensions could give a follow up on how well the schedule is kept. I.e. which are currently active? Which have been put on hold?

More follow up is needed in a final version. In the current version it is not possible to see if the chart is going as planned. One cannot be certain that the time schedule is followed. Support for checking if there are delays and how the schedule is affected should be a part of a final version.

The current schedule is planned after the critical path algorithm. The critical path algorithm gives the smallest possible makespan, i.e. shortest total duration of the project. However, this is not always the most realistic schedule. For example, there might be vacation days or someone might get sick. These scenarios must also be taken care of before a release of the product.

The timeline is also in need of some work. Since all tasks are always fitted into the width of the page, there is a problem when there is a task with a very long duration and one with a very small. In comparison, the task with the small duration will take almost no space at all while the longer task will take a lot of space. In the current version, the time interval is always 5 days. A more reasonable solution would be to have a dynamic timeline, which might depend on the overall length of the project. Projects that have a duration of less than a month might be shown on a weekly basis, project with a duration between a month and a year could have a monthly time axis. Perhaps one could also add a feature to zoom in or out to switch between the different settings.

5.4 Work distribution

The design process have been conducted together between us. For the implementation, Denhi focused mostly on the genetic algorithm and task assignment extension, while Nguyen figured out the visualization with the Gantt chart.

6 Conclusion

In this project, a planning support tool has been developed where tasks in a software project are assigned to different persons of the project. The planning support tool has been developed as extensions to Qlik's software Qlik Sense.

Two algorithms has been implemented for this project, the critical path algorithm and a genetic algorithm. The critical path algorithm calculated the start and end date of each task based on the dependencies between different tasks. The genetic algorithm is an algorithm which mimics the natural evolution process and was used to calculate a population of possible solutions for the person to task assignment based on three objectives, namely match rate, workload balance and task overlap. From the population, the solution which best fulfilled all objectives was selected and visualized in the task assignment extension.

Two extensions were created to visualize the results from the algorithm, a Gantt chart and a task assignment extension. The Gantt chart consists of tasks put horizontally on a timeline of execution. The task assignment extension is based on D3JS' partition layout and shows the correspondence between the persons and all assigned tasks.

The extensions were developed using an agile development process called SCRUM. A design process was interlaced with the development process which focused on user centered design. The design process consisted of interviews, brainstorming, lo-fi prototypes, hi-fi prototypes and a user study. Two user studies were conducted. The first user study was based on four users, two beginner users and two expert users. After the first user study, some improvements were made before the next iteration of user tests, which was tested on four additional intermediate to expert Qlik Sense users.

The end product created in this master thesis is a prototype and work in progress. The planning support tool focused on software development projects, but could be extended to be used in any line of business.

References

[1] Qlik's company history, [website], http://www.qlik.com/company/about-the-company/history, (accessed 29 January 2016).

[2] Download of Qlik sense, [website], http://www.qlik.com/try-or-buy/download-qlik-sense, (accessed 29 January 2016)

[3] Arvola,M., *Interaktionsdesign och UX*, Lund, Studentlitteratur AB, 2014.

[4] Floyd, C., Budde, R., Kuhlenkamp, K., Mathiassen, L., & Zullighoven, H., A systematic look at prototyping, Approaches to prototyping, Namur, Belgium, Springer-verlag, 1983, pp. 1-17.

[5] Tullis, T. and Albert, B., Measuring the user experience, Burlington, MA, Morgan Kaufmann, 2008

[6] Nielsen J., How Many Test Users in a Usability Study?,[website], https://www.nngroup.com/articles/how-many-test-users/, (accessed 24 May 2016).

[7] Ziemkiewicz, C., Kosara, R., The shaping of information by visual metaphors, [website], 2008, http://kosara.net/papers/2008/Ziemkiewicz_InfoVis_2008.pdf, (accessed 1 February 2016).

[8] Norman, D. A., Affordance, conventions, and design, [website], 1999, http://webhome.cs.uvic.ca/~mtory/courses/hci_spring2007/references/norman.pdf, (accessed 4 February 2016).

[9] Norman, D. A., The design of everyday things, USA, Basic books, 1988

[10] Baumann, K. and Thomas, B., User interface design of electronic appliances, London, Taylor & Francis. 2001.

[11] Balaji, S., & Murugaiyan, M. S. (2012). Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. International Journal of Information Technology and Business Management, 2(1), 26-30.

[12] Beck, K., Embracing change with extreme programming, Computer, vol. 32, no 10, 1999, pp. 70-77.

[13] Schwaber, K. and Sutherland, J., Scrum, [website], http://www.scribd.com/doc/35686704/Scrum-Guide, (accessed 1 february 2016)

[14] Faced with a complex project? Use Scrum to improve teamwork, communications, and speed, [Online video], 2016, https://www.scrumalliance.org/why-scrum, (accessed 1 february 2016).

[15] Kokash, N., An introduction to heuristic algorithms, Department of Informatics and Telecommunications, University of Trento, Italy.

[16] Wall, M. B., A genetic algorithm for resource-constrained scheduling, Doctoral dissertation, Massachusetts Institute of Technology, 1996

[17] Gantt chart history, [website], http://www.gantt.com, (accessed 15 May 2016)

[18] The invision Hi-Fi prototyping tool, [website], http://www.invisionapp.com/, (accessed 17 February 2016)

[19] D3JS, Data-driven documents, [website], https://d3js.org, (accessed 21 april 2016).

[20] Dix, A. Finlay,J. Gregory D.A. Beale R., Human-Computer interaction, US, Springer, 2009, p 38.

Appendices

Intervju

Frågor

- Vilken information söker du efter när du planerar?
- Vad anser du är svårast med planering?
- Hur tar du reda på att du behöver omplanera?
    - Hur går du till väga om du behöver omplanera?
    - Hur ofta behöver du omplanera?
- Hur bedömer du, hur du tilldelar personal till olika projekt?
    - Hur bedömer du ifall någon är lämplig för ett visst task/projekt?
- Använder du planeringsverktyg när du ska tilldela personal till olika projekt/tasks?
    - Om ja, i så fall, vilket/vilka?, fördelar?, nackdelar?
    - Om nej, Vad använder du istället?, fördelar?, nackdelar?

- Vilken funktionalitet önskar du till ditt planeringsverktyg/ideala planeringsverktyg?

Usability test of a planning support tool

Background

For this test case, you will be exposed to a planning support tool in the prototype stage. The tool is integrated in Qlik sense and our objective is to evaluate its usability. You will be asked to perform certain tasks in the tool. The tool comprises of two already predefined extensions.

You are a project manager for a software project. To assist you in making decisions you are using the Qlik sense software with an extension to help you visualize tasks that are assigned to persons and another extension showing when the tasks should be performed.

In this tool, you will be introduced to an example software project. The project consists of different tasks which has a start date, end date, duration, and dependencies. A dependency means that a following task can not be performed before the previous task is finished. The dependencies can belong to a critical path or a non critical path. A critical path means that if a delay is made in task with a critical dependency, then the entire project will be delayed.

The project also has persons included in the project. The persons can be assigned to perform one or several tasks.

Consent

I hereby agree to voluntarily participate in a usability test of a planning support tool in Qlik Sense. I am informed that my personal details and results will be handled confidentially and will be presented anonymously. I understand that information about the tool given by me in this usability test can be used without further notice. I am aware that I can at any point cancel the session.

June 1st 2016, Lund, Sweden

Name:

____

Signature:

____

- Give examples of tasks that are currently in progress.

- Determine the start date and end date for a task.

- Determine the duration for a task.

- How many days remain on the project?

- Which tasks are already finished?

- Find out which tasks that belong to a critical path.

- Find out which person an arbitrary task belong to.

- Determine which tasks that belongs to an arbitrary person.
- When are these tasks scheduled to be performed?
- Return back to the overview with all people.

- Find all persons which belong to the same task chain.

- Which features do you like?

- Which features do you dislike?

- Which features do you miss?