Final Report

21/05-11

Group I

Dmitry Igoshin
Kirill Blazhko
Johan W Schutzer
Sebastian Hanson
Mahsa Mirtalebi
Ali Nazar

• Project background

The aim of this project was to create a GPS system with navigation features.

In this project we were four individuals, including myself, who have been working together before. I think this was one aspect that helped us succeed with our set up tasks, since we already knew and understood each other.

In the beginning of the project, we thought about implementing the GPS system for an android smart phone. It sounded exciting since the product might be useful, compared to a GPS system on a laptop. However, after a sort session of researching and testing the software tools for android development, we decided to implement it for a pc environment instead.

When we had settled for the pc, we decided that this product should be discrete, solid and fully working. With this we meant that instead of implementing a big application with a lot of features that might work, we wanted to implement a not so big application, but with higher quality instead. We wanted it to be memory efficient, fast and user friendly.

• Project organization

For our project we used scrum as the development methodology. This methodology was in fact a requirement specified by the course coordinator and was mandatory to use. The person that was assigned to be scrum master had used Scrum development methodology in his previous project so he was a bit familiar with the development methodology.
As scrum suggests, we were allocated 8 sprints, also predefined by the teacher. Sprints basically allowed us to develop iteratively and almost produce software each week. In the team we decided that we were going to have daily scrums at least each project working day. Each of the daily scrums has been thoroughly documented. During the first implementation days we decided that we would work from 9 in the morning to 17,00 because we had tightly scheduled lessons, however after some weeks we decided to work from 10 in the morning to 17,00 each day. Due to a good organization within the team and using the daily scrums to organize and keep the team working together and not working in separately.
In the team basically everyone had the role of developers however we assigned the scrum mastery to Ali Nazar.
We created a product backlog and for each sprint we created a sprint backlog in which we described thoroughly what every developer's tasks were.
Because of the fact that we were friends from before and working in the same development team, the team respected each other and followed the plans written down by the scrum master. So the organization, through this became very solid.

- ## Project Requirements
  Specific Requirements

In this section, all of the requirements that were created during the requirements elicitation process are listed. Here we give comments about how each requirement worked out in the final product.

- **The software should have cross-platform support.**
  -The software now works for both Windows & Linux.

- **It should be able to get the current location.**
  -This was one of the first thing we focused on after connecting the gps, we parsed the information and got our current lat & long variables.

- **It should be able to handle LOC files for coordinates, to get the target location.**
  -We now have a feature based on a button connected to a filechooser, just load your file and follow instructions.

- **It should show a map, N/S W/E calibrated.**
  -The map static and will not rotate, but north is always up and south always down on the screen.

- **The map should include a zooming function.**
  -Zoom function was easy to implement becouse it was a part of the osm api we used.

- **The map should include a scrolling function.**
  -Scrolling function was also included in the JMapViewer and it was easy to implement.

- **It should include a GUI with at least zooming and scrolling buttons**
  -JmapViewer provided us with these tools, very easy to implement.

- **It should give a notification when the target is reached**
  -We put a boolean variable to true if we are within 20meters of target location, a notification box appears and inform you that target is reached.

- **It should show where the current location and target location is on the map, graphically.**

-We have red map marker dot showing our current location, a blue dot for target location and green lines in between showing the shortest path. Current location dot and the lines is always updating.

- **It should give a notification when the user is heading in the wrong direction.**

-This is not implemented because we had some problems with the closest node sometimes showing that your supposed to go back to then continue forward.

- **It should update the users current location at a regular basis**

-Current location is updating every second.

- **The guidance should be one line from current location to target location.**

-We have a green line between the two locations showing the closest path between point A and point B, taking obsticles into consideration.

- **With the updated current location, it should update the suited way to the target location.**

-Same as above

Non-Functional Requirements

- **It should show the distance**

-We have a class for calculating the distance between current location & target location, this is displayed in the GUI info panel.

- **It might have sound indications.**

-We still don't have sound indications, might implement it before release.

- **It should have a Colorful and transparent GUI.**

-We where very attentive when designing the GUI, taking in to consideration that we wanted to have a Simple, user friendly and memory effient GUI, and i think we succeded with that.

- **It should have different modes (fastest way, nicest way, easiest way.**

-This was not possible to implement before deadline.

- **It should have a save/load function for routes**

-We created our own fileformat called .route, we can both create new routes, save them and then load them from a list.

- **It should be able to estimate the time needed to get to the target.**

-Not implemented because pathfinding sometimes want to navigate backwards to the closest node.

- **The user should be able to input a target location.**

-The user is able to input any adress in the world and location dot is shown on the map. Though pathfinding to the location can only be done within the limits of Lindholmen, outside lindholmen the line will show the bird way.

- **Multilanguage interface.**

-Not implemented.

- **Compass**

-We did work on a compass for our gps, showing wich direction we

where heading, not working correctly at the moment.

- **Networking between two computers so that both can see coordinations on the map. Chat function/guidance function.**

-Now implemented, you are able to see your friends location and chat between computers, still abit laggy but works.

- **Showing some areas of interest on the map by a video, the video will be showing the surrounding area and the area of interest**

-We can display and undisplay interest points on the map, we have (Atm's, Parking, Restaurants, Buss Stops).

- **caloric burnage (using speed parameter).**

-Not Implemented.

- **Implement a self-rendered map.**

-Not implemented, we are using a online based api map from open street maps.

- **Connect an XBOX joystick, so that the maps position can be changed using the joystick.**

-Not implemented.

- **The application should be connected with twitter and facebook, you should be able to update your wall/tweet your location. This is to make the program more personal.**

-We now have a twitter function, you send a message to twitter and your current location by a link to google maps.

-

- ## Lessons learned

throughout work with the project we have gained a lot of experience.

Here are some learnings that probably will help us in our future work:

1. Git is not fully compatible with Eclipse:

Our guess is that Eclipse has its own cache from which it fetches data time to time. Thus data conflicts consistency are introduced. There is also a problem with capital letters in the names of the classes. One should be very cautious when naming the classes in camelCase and committing those to the repository.

2. Hand-written Java Dijkstra's algorithm has sufficient performance only with 'one-person' use case. It is slow for many near-simultaneous shortest path queries.

We have overcome this obstacle by precalculating all the possible shortest paths and saving them to a database.

3. Ruby and PHP parsers and is easier to maintain than Java parsers.

We had to write a few parsers for .osm files and we found out that it is much faster and easier to have small server-side PHP or Ruby scripts which do a pretty fast stream parsing of the map data files.

4. MySQL-based routing is slower than PostgreSQL+PostGIS+pgRouting.

We found that only in the end of the project term, but this knowledge is really interesting: thanks to new geometry types and operators PostGIS allows to create really powerfull (in terms of number of shortest path queries) applications.

• Organization issue: in the beginning, each task was assigned 100 points. This was wrong, because the tasks have different difficulty. Now each task has a number from Fibonacci sequence to reflect the difficulty.

• GUI issue: GUI was taking too much memory. After some research, and using experience from the previous project, we found out how to delete graphical objects from memory.

- ## Future outlook

The experience we have gotten from working with each other has been, according to each team member in the team, one of the best working experiences so far. With this said, we would really consider working within the same team for the next upcoming projects. The friendship and comradeship in between team members has been explicitly good and this just made the whole team organization even better. However, there are some team members that would like to join another team, just for the sake of more and newer experience, but still they are very willing to work in the same team for the coming projects.
By now we now our strong and weak points and besides being team members we have also established a good friendship between each other. Our opinions was basically that it would be very good if the team stick together for the coming projects, however if there is anyone in the team that would like to join another group, he / she is most welcome to do so.