Ragger Jonkers - 10542604
Ellen van ‘t Klooster - 10207309
Belle Bruinsma - 10676759
Urscha Fajdiga - 11377437
Edwin Steffens - esteffe1

Part 1 - Analysis

Think about how these networks are different. Analyze the “dimensions” of these networks. What are the relevant attributes (e.g., commits, users, branches, commit size, etc.) of these representations? What other attributes could be relevant in this graph? Write a list of all the attributes your visualization could show.

In the network visualization of github we can see the different users and their commits on a -what appears to be- non-linear time scale. Also it is possible to see the connection between the users and their commits between branches (displayed with arrows). What seems to be lacking is the size of the commit, which we think we could visualize in lines of code added. The last attribute is probably very useful, because people will look at this graph to get an impression about who did what and how the project developed. If someone is only editing spelling mistakes in every commit, this should be visible in some kind of way. The interactivity of being able to drag along the time axis is a nice touch, and it would be useful to have the same or a similar approach in another type of visualization..

Are there different roles, i.e., different types of users who might want to achieve different things? Write a list of user roles.

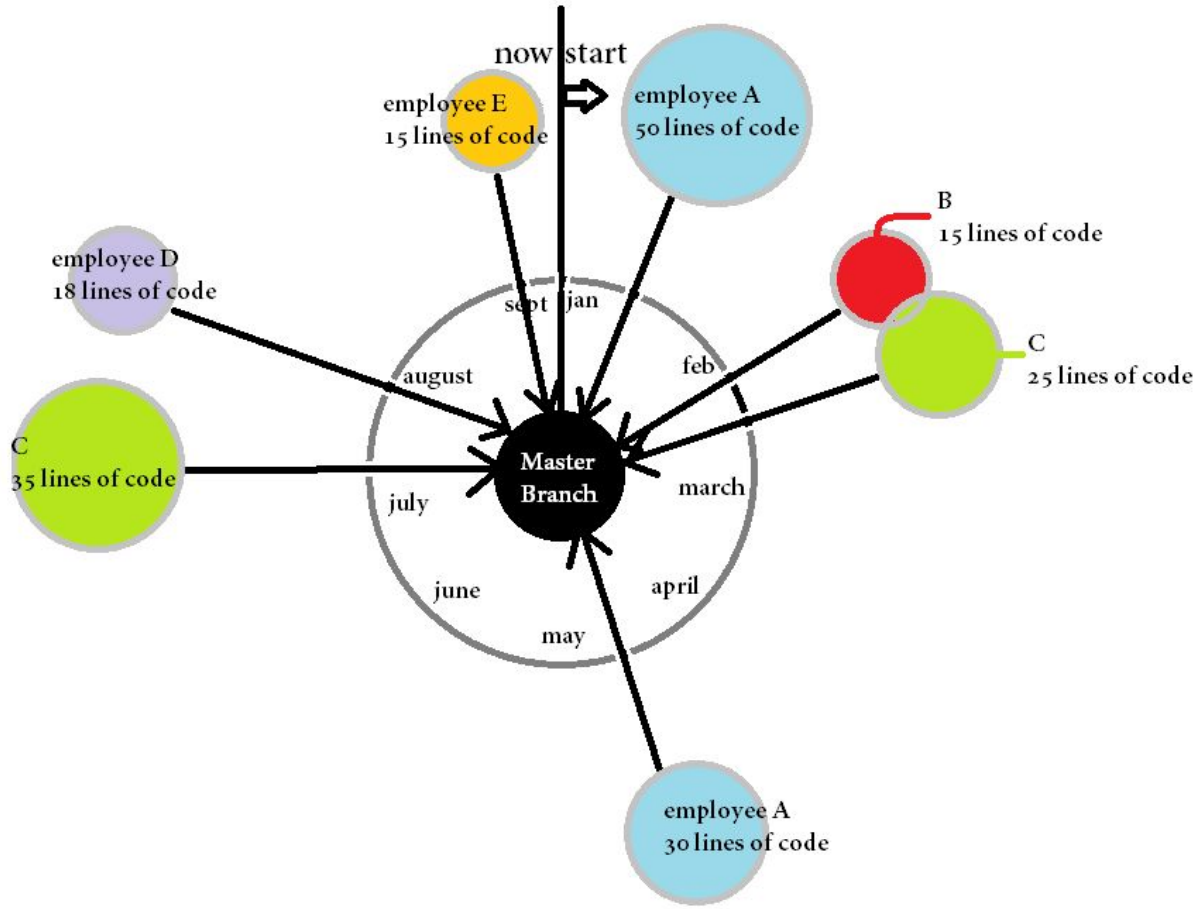
The role as collaborator is mainly to quickly get an idea of what your other collaborators have been / are working on. The role of a project manager is different. He/she should be able to see where things stagnate, so for them it might be important to get a clear indication at which point in *time* certain things did or didn’t happen. Outsider roles of for example recruiters might be to find out the head developer (or most contributing/important) of the project by looking at the amount of commits (and commit size).

Identify one role that you want to design your visualization for. Prioritize your task and attribute lists based on this role’s needs.

We want to design a visualization for the project manager. He/she should be able to easily see who has been working on the project recently and how fast the process is going. This role needs: to quickly get an overview of how the process is going with a quick glance, see who has been working hard and who is a free-rider on the team of programmers, get a good sense of the time-frame and the pace of the project moving forward.

Part 2 - Sketching

Visual representation 1:



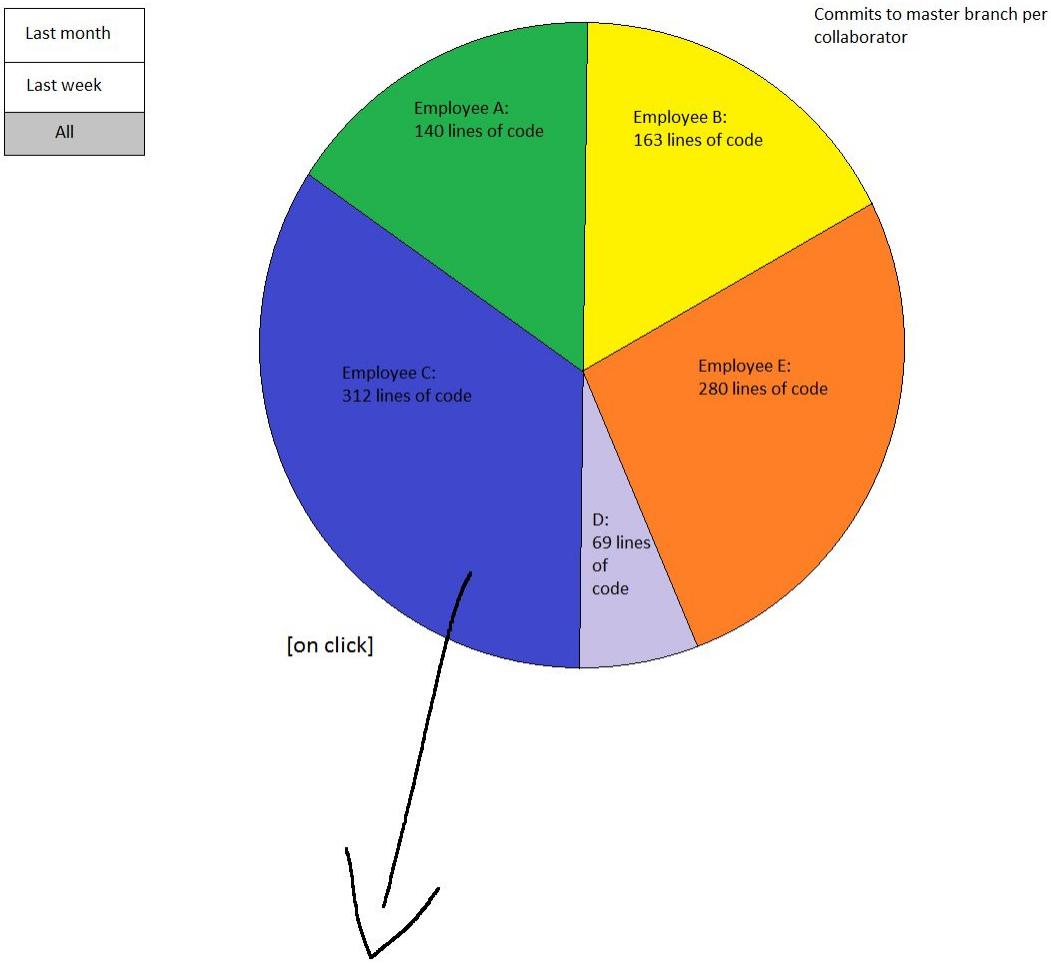
Visual variables:

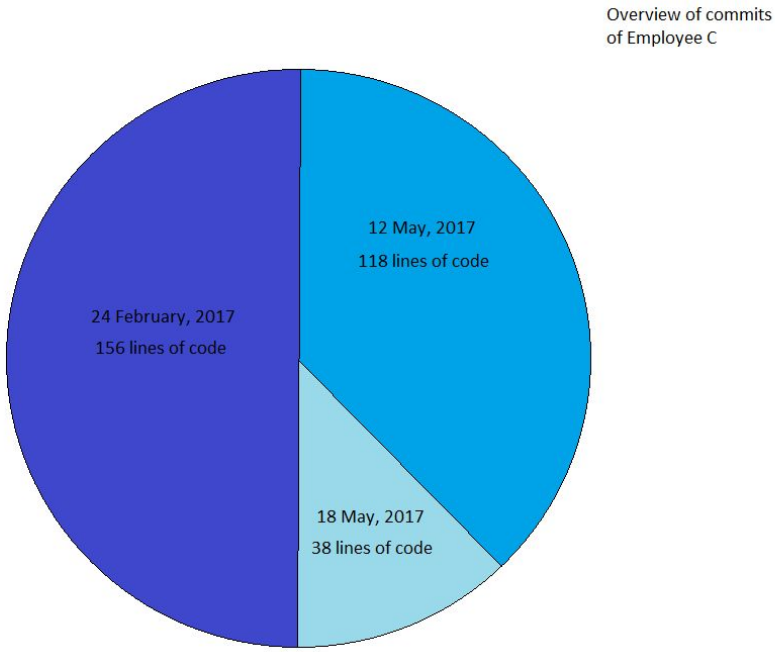
- Color: we use a different colour for every employee (or Github username) to easily show who is who, and who committed lines of code to the masterbranch.
- Size: the size of the nodes of the commit represents the lines of code that the employee submitted to the masterbranch. So if an employee has more and bigger nodes, he/she has submitted the most often and the most lines of code.

Every commit is shown here as a single node because this gives the project manager the best overview of all the commits that are made over a certain time-period (that is shown on the inner circle). The time frame is linear in a sense that there is equal space between the months, even if there are no commits during that period. If the project is bigger however, with a lot more commits, showing every commit on a separate node, could be chart-junk and the graph would not be as clear anymore. So this node-link graph is appropriate because it gives the project manager a lot of information very quickly, namely who has committed lines of code, how many lines of code and when these commits have happened. But when projects become larger, an alternative graph could be more useful.

Visual representation 2:

In contrary to more visually advanced graphs the pie chart is one of the most simple and elegant charts to quickly visualize the proportion of work someone has done in comparison to their colleagues. The chart would be able to dynamically change based on which point in time the user wants to look forward from.





Visual variables:

- Color: we use a different colour for every employee (or Github username) to easily show who is who, and who committed lines of code to the masterbranch.
- Proportion/ratio: the amount of lines committed defines the corresponding share of the pie chart for that user
- Size: It would also be possible to have multiple pie charts each of a different branch. The size of the masterbranch pie chart could be made relative to charts of other branches taking as scale the total amount of lines of code committed per branch

Reflection

1. Target: choose a specific domain define research question(s) find & clean the data.

In the first part (analyses) of the assignment they let us think about the different dimensions of the networks at the given examples. We choose the role of a project manager because this role needs: to quickly get an overview of how the process is going with a quick glance, see who has been working hard and who is a free-rider on the team of programmers, get a good sense of the time-frame and the pace of the project moving forward.

2. Translate: formulate data analysis tasks exploratory data analysis transform & summarize data.

We discussed which data was important and wrote the data down of which employee wrote how many coding lines.

3. Design: design visual encodings design interactions sketch many ideas!

First we wrote some things down who were important according to us when a project manager needs some quick information. We used color, size and ratio to make some sketches. First we made a chart-junk and showed every employee with a bar in a different color with their coding lines. We also wanted to show that the time frame was linear in a sense that there is an equal space between the months but this didn't look as clear anymore. That's why we made a Node Link Graph. Now every single node was an employee in a different color with a different size of node and the time frame is shown. The downside is when the projects become larger it won't look as clear anymore. That's why we sketched a Pie Chart. This is a very clear and clean chart and shows the proportion of work someone has done and when, in comparison to their colleagues.

4. Implement: use code "sketches" define data structures find efficient algorithms.

This was the part that we used the sketches and made the Node Link Graph and the Pie Chart.

5. Validate: is the abstraction right? does it support the tasks? does it provide new insights?

This was a design for the project manager, so he/she would be able to easily see who has been working on the project recently and how fast the process was going. So she need a quickly readable overview of how the process is going, see who has been working hard and who is a free-rider on the team of programmers, get a good sense of the time-frame and the pace of the project moving forward. We think the Pie Chart is the most abstract and clear. This is because you see in one glance who wrote the most lines of code. If you click on a specific employee you'll see the days they wrote the number of lines on. So you see the most important information about the employees in the Pie Chart and if you want to see more specifics the project manager could see it by a click. The Node Link Graph contains a lot of information so you first have to look at it for a second. But then you have all the information about who wrote the most lines of code and when.