Team Reflection

Week 40

Team members: *Nur Hussein Abdulkader, Tom Bjurenlind, Daniel Cebe, Rickard Gyllensten, Roman Melnik & Christer Sonesson*

Customer Value and Scope

- the chosen scope of the application under development including priority of features and for whom you are creating value

  The application is supposed to give points for the number of passengers in a car. The points can be accumulated in order to earn rewards such as coffee or snacks at gas stations. The idea is to encourage carpooling in order to reduce air pollution and traffic congestion. The priority of features: identify the number of people in the car, award points for the current ride, store/load points and show total, create a shop where rewards can be redeemed. Value created for: end consumers (drivers), Volvo Car Corporation, the society (less pollution, less congestion).
- the success criteria for the team in terms of what you want to achieve with your application

  A user-friendly and intuitive application that gets many active users who carpool often, thus earning a lot of points and redeeming a lot of rewards. A cleaner environment and less congested roads.

- your user stories in terms of using a standard pattern, acceptance criteria, task breakdown and effort estimation
  - As an eco-friendly driver, I want to be rewarded for driving with several people in the car and possibly for e.g. driving with low RPM where possible (high gears).
  - As a driver I want to see my awarded points on the board that shows the number of points awarded for the current trip and in total.
  - As a Volvo driver I want to be able to visit the store page quickly, since I want to spend my points for rewards, and view my inventory (what I have spent my points on).
  - As a team we want to learn about Android API since we need this information to to complete this project
  - As a team we need a version control system git repository so we can communicate and work with the tasks
  - As a developer I want to use the simulator signals in order to determine the number of people in the car.
  - As a driver I want to see the items I have bought in an inventory since it's easier to visualise the items.

Our general strategy for handling the above user stories is to attempt to estimate how much effort they require in terms of one Sprint (one week's task work) and break them down accordingly to

smaller features, and these into even smaller tasks; which may then be individually completed during one or two week's work (approximately). Our acceptance criteria is simply that the user story is met by our implementation. If we run into some complication with this criteria in the future we'll have to try to compromise somehow.

- your acceptance tests, such as how they were performed and with whom
  Acceptance tests for:
  - Creating 'Home' and 'Back' buttons in app: testing that the buttons work as they are supposed to.
  - Getting simulator signals from the seat belt sensors: testing that the signals are read as they should and that the number of people in the car is calculated, used and displayed.
  - Scoreboard for the current trip: visually verifying that the scoreboard shows the correct number of point for the current trip.

- the three KPIs you use for monitoring your progress and how you use them
  We have now developed a few features and our main KPIs are still team velocity and effort estimation. We want to keep a reasonable pace and complete the tasks assigned for the current sprint. Practically the way we measure our work is by how many tasks and/or how much on each task we complete each week. Next week we probably want to do slightly more task work than this week, but about the same should do, we think. To do this everyone in the team has to actively contribute to the deliveries with taskwork.

Social Contract and Effort

- your social contract, i.e., the rules that define how you work together as a team (this means, of course, you should create one in the first week)
  Our social contract still hasn't changed since formulated the previous weeks. We still strive to help each other out when there's a need as well as divide the work equally between all team members. We will try to follow this contract as far as possible. If we'd discover further ahead that something isn't working with respect to this contract, then we'll have to tackle that problem appropriately when (if) it does happen, and perhaps modify the contract.
- the time you have spent on the course (so keep track of your hours so you can describe the current situation)
  Like previous weeks we define the hours spent on the course as the hours spent strictly on the project (in group work plus mean individual work at home). With this definition we have probably spent around 12 hours this week, so in total 8 + 13 + 14 + 12 = 47 hours (as previous weeks' work was estimated to about 35 hours). Among this week's hours, about 1 hour was spent on the week's counseling meeting, 7 hours on a collaborative mid-week meeting and the rest on individual task work from home (the mean over each member's individual work). (The time for writing these reflections is excluded in this time estimate as per the usual.) We estimate that the next week's hours spent should be about the same; as we continue working off the

remaining tasks (and possibly come up with new ones). If more hours is demanded from us in order to finish our tasks for the next sprint then we will have to try to put in more hours collectively as a team.

Design decisions and product structure

- how your design decisions (e.g., choice of APIs, architecture patterns, behaviour) support customer value

  This week we continued on implementing the actual app (task work); so we got further experience with the different APIs. We have only used the standard Android APIs and Android CAR APIs thus far still, and we have built on top of the foundation work we executed previous weeks. This week there were definitely more focus on good design decisions; some of the foundation work from previous week was updated to reflect this. Next week we continue to work along these lines; with good design decisions in mind. To do this we will have to continue having good design decisions in mind when pushing new features; and refactor existing work where appropriate.
- what you document and why, by using e.g. use cases, interaction diagrams, class diagrams, domain models or component diagrams, text documents etc.

  Similar to last week we have only documented our project by our Scrum planning-board (Trello), drawn images of the different app screens on paper sheets, written reflections in text documents, and commented our code in an intuitive and explaining way. We'd say this works well for us still; so we do not really have any plans of incorporating further documentation (such as UML diagrams) next week neither. If we find out later that we might need further documentation for whatever reasons, then we will have to come up with an appropriate solution when/if that happens to be the case.
- how you use and update your documentation throughout the sprints

  Same as the week before. We have been using Trello to set up tasks and user stories, and we continuously update this as we progress. Google drive is used to write all reports, which are then uploaded to github. Github is naturally also used to store the project and all code. This is a process that works well and one that we plan to continue using.

- how you ensure code quality, enforce coding standards, and application of scrum

  No major changes from last week. Each team-member is responsible for his own code, but cross-checking by other team-members is also used, so any deviations from the team-standard can be caught. Github is used to store the code, and with that all changes can be easily reviewed, and we can pull back changes to the code if necessary. A lot of code is produced in group-sessions, where most of the team can get together and solve coding problems (and review code), leading to a higher coding standard. Scrum is enforced partly by the scrum-leader, and partly by the will of the team. Everyone makes an effort to learn and apply scrum to the development process. This works well at the moment and it is a process that we plan to continue using.

Application of Scrum

- the roles you have used within the team

  The roles for the team are the same as the previous week. We have a scrum master who negotiate with other teams and ask/communicate with the product owner if needed. And as before the remaining five team members are ordinary developers. Their job is to contribute something to the team in the form of develop the product by encoding and complete the tasks. The scrum master also helps to develop the product as mentioned last time, but he also make sure that everything is fine and if there is some questions to the product owner or general questions to other teams.

  Further we should continue to use our roles as they are and continue to develop and contribute with everything that is required to deliver an application. To get there every team member need to follow the roles. As a developer for example, namely to develop code and be involved in communications and to perform the work that is relevant, specific, for every week. And finally in the end deliver a product.

- the agile practices you have used for the current sprint

For the current sprint we have used user stories and completed some of the tasks that are related to the user stories. We've had stand-up meetings one time this week and also one meet for 4 hours to discuss and start encode. We have tested our progress continuously, and we try to keep a sustainable pace in our work.

  Compared to last week we have start encode and also complete several tasks that requires encoding. Moreover we have to continue encode with the significant parts of the project and continue having stand-up meetings online or physically and include more testing in our progress.

  To get there we have to work efficiently, divide code into small portions, test often and set up more meetings to achieve the best performance for the sprint. So for next week the goal is to meet first at monday and then at wednesday and encode.

- the sprint review (either in terms of outcome of the current week's exercise or meeting the product owner)

  We meet with the product owner from Volvo during mondays supervision once again. He got some good feedback on the work and idéas we had so far. And he also shared some interesting paths we could take in the future to improve the application. We did not attend any exercise this week but we did get together on wednesday for several hours to work on the application. This lead to plenty of tasks being done and a lot of final value delivered.

- best practices for using new tools and technologies (IDEs, version control, scrum boards etc.)

  For this sprint we really got the tools working together like a machine.The emulator in android-studio was working without problems and with simPy we managed to do our

first simulation. The group also completed more scrum-tasks than ever. Using Github as our VCS starts to feel more natural and everyone is working in their own branch and committing to the master-branch. Communication is still mostly done over messenger but also on our own discord server, which gives us the ability to both write and speak with each other.

- relation to literature and guest lectures (how do your reflections relate to what others have to say?)

  Our reflections relate to what others have to say structurally in the sense that we use the A-B-A->B way of reflecting where we find it relevant and/or applicable; where A is where we are currently, B is where we want to be next week, and A->B explains how we might get from A to B. When it comes to what we actually say when we reflect in this way (the actual content), we have not really read any reflections of others this week neither, so we couldn't really say how it relates to what others say.