Student Feedback

Project CS 2016

Authors:

Andrew Aziz
Andrew.Aziz.2356@student.uu.se

Ludwing Franquiz
Ludwing.Franquiz.2071@student.uu.se

Max Wijnbladh
Max.Wijnbladh.2971@student.uu.se

Eric Wang(Yiqing)
Yiqing.Wang.6263@student.uu.se

Maria Rajabzadeh Namaghi
Maria.Rajabzadehnamaghi.5780@student.uu.se

Theodosios Malatestas
Theodosios.Malatestas.1471@student.uu.se

Johan Snider
Johan.Snider.3503@student.uu.se

Sami Kaivonen
Sami.Kaivonen.4995@student.uu.se

Adrian Amigues
Adrian.Amigues.7647@student.uu.se

Eirini Petraki
Eirini.Petraki.5059@student.uu.se

Uppsala University

January 7, 2017

1 Feedback from whole team

First of all on the behalf of the whole team we would like to thank Uppsala University, Ericsson and SICS for the opportunity of being a part of a real industrial project. It was an experience that we all learned from. Researching an up-to-date topic such as ICN-IoT gave all of us insights into a technology that will be present and relevant in the future tech world. We all gained experience working on an industry level project from planning to realization. For example, we gained experience in planning how to design a system, how to work in a bigger team than must of us were used to and using multiple programming languages in one project.

Another important thing we learned throughout the course was to work within a team of people from different backgrounds. In the beginning of the project, we had some debate on how to begin with development since people with different backgrounds had been taught differently. And throughout the course we had some disagreements about how certain design choices and planning. Through discussion we were able to compromise and continue in the development process.

We would also like to thank Uppsala University and Edith Ngai for their efforts in organizing the project and for the two rooms that were reserved for us during the entire course. We felt that having our own space majorly added to the feeling of working in an office and to the group dynamics, especially the lunch room which we used as well.

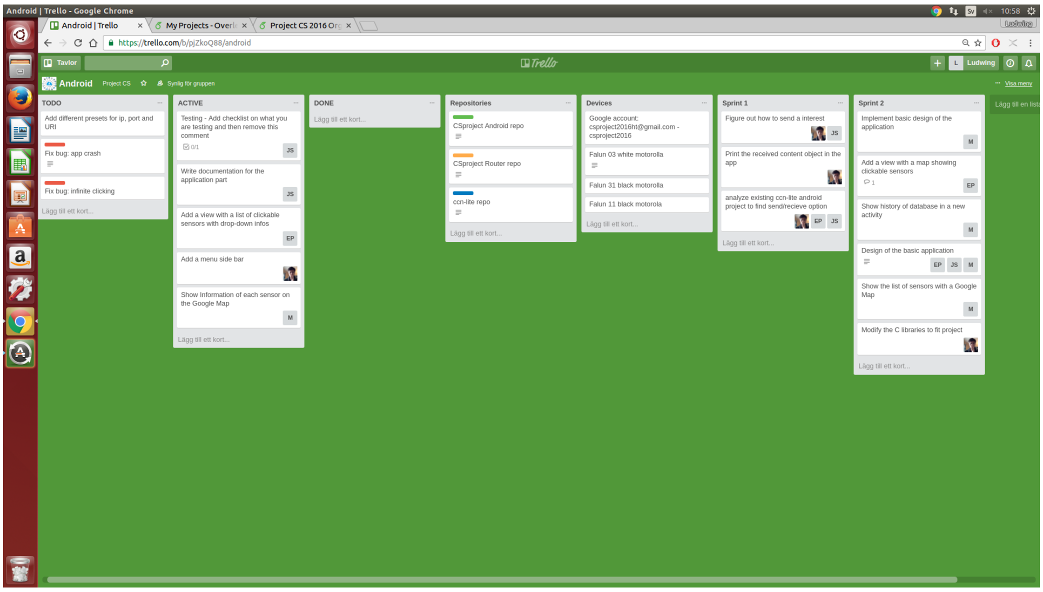
The availability of the industrial partners and the meetings with them helped us stay focused on our tasks and provided us with crucial feedback during the course. For example, when we had problems with software or when we had problems to understand how they wanted a specific deliverable to be developed they were always there to answer. The meetings also helped us during the design phases of the project since the industrial partners could tell us quickly if we were on the right track or not.

We would also like to offers thanks to Mikael Lundgren who came in during the first week of the project to lecture us and answer questions we had about agile software development. We would also like to thank Markus Bylund and Olle Bergdahl from the Uppsala Kommun who came in and gave us a presentation about how our work could be applied to in the Uppsala Kommun and what other projects the Kommun is working on.

1.1 Scrum

To organize the work in the project, we decided to use the agile methodology Scrum. In Scrum the work is divided into sprints, which are typically two week intervals in which some amount of work should be finished. The first part of the sprint is to have a planning meeting where the team sets the length, tasks and objectives of the coming sprint. In this meeting the team also decides how to divide the different task within the group and what needs to be accomplished to have a working minimum valuable product by the end of the sprint. One of the last topics that is discussed in a sprint planning meeting is about deciding on who will work on which part during the sprint. Additionally, during the sprint there are usually short meetings in the morning the recap on the goals of the sprint and keep an eye out for any problems that might arise.

We used a software tool called Trello for organizing our Scrum objectives. Trello is a tool designed for agile development and offers all the features you need when working in Scrum. To illustrate an example of how we worked, figure 1 shows the board for the development of the Android application. In the first list to the right are all the future tasks. These are things that will have to be done at some point, but that aren't active at the moment. Next is the list with active tasks showing the people that are working on them. When the active tasks are done, the cards are moved to the list named done. There are also lists showing the repositories and devices that are used for the Android application. Finally, there are lists for the different sprints where the tasks done during the specific sprint are collected, this allows the team to have a good overview of what has been done during the sprints throughout the project.



TODO

- Add different presets for ip, port and URI
- Fix bug: app crash
- Fix bug: infinite clicking

ACTIVE

- Testing - Add checklist on what you are testing and then remove this comment (0/1, JS)
- Write documentation for the application part (JS)
- Add a view with a list of clickable sensors with drop-down infos (EP)
- Add a menu side bar
- Show Information of each sensor on the Google Map (M)

DONE

Repositories

- CSproject Android repo
- CSproject Router repo
- ccn-lite repo

Devices

- Google account: csproject2016ht@gmail.com - csproject2016
- Falun 03 white motorolla
- Falun 31 black motorolla
- Falun 11 black motorola

Sprint 1

- Figure out how to send a interest (JS)
- Print the received content object in the app
- analyze existing ccn-lite android project to find send/recieve option (EP, JS)

Sprint 2

- Implement basic design of the application (M)
- Add a view with a map showing clickable sensors (1, EP)
- Show history of database in a new activity (M)
- Design of the basic application (EP, JS, M)
- Show the list of sensors with a Google Map (M)
- Modify the C libraries to fit project

Lägg till ett kort...

Figure 1: Trello board for Android team.

A key idea in Scrum is that each team member should be able to work on different parts of the system during the project. This is because every team should know about what the other teams are working on from the daily morning meetings. Another important element in Scrum is how fast the group can react to changes and problems. Because the objectives and goals are tied to a sprint, if there are any major changes the goals and objectives can be reevaluated in the next sprint. At the end of each sprint a retrospective is conducted, where each member can share their opinion on what went well during the sprint and what can be improved to the next sprint.

For us, the decision to use Scrum was a key factor in being able to organize well within a group of eleven students. In the beginning of our project Scrum gave us an organizational framework for us to use. Since many of us did not have any experience within the scope of the project. Scrum also allowed everyone to be apart of all the parts of project and not as in previous projects where you just have had understanding of one part. The minimum valuable project that we showed Ericsson and SICS after each sprint was helpful since they could see our results and give feedback on our progress.

1.2 Advice to staff

We think that administrative part of the project was well handled by Edith and the TAs such as ordering equipment, having the individual meetings and taking attendance. Also something we really appreciated was the lectures and workshops in the beginning of the project to give us a good starting point for the project.

It was unfortunate that the TAs for the course didn't have any specialization or focus in the topics of ICN or IoT. For next year we would recommend that the course should have TAs which have knowledge that fits the topic of the course. In retrospect, it would have been helpful to have some day-to-day help working with the sensors and other hardware used in the course, especially in the first four to six weeks of the course.

1.3 Advice to students

The project CS course is a preparation for the thesis course and work. Beside the opportunity to improve technical skills, the course gives the opportunity to work on soft skills needed in the workplace. It also gives you a opportunity to learn and experience direct contact with a real customer which in our case was Ericsson and SICS. This also gives you contact with industrial partners which might make it easier for you to get a thesis or work at their company. However, if you as a student are thinking about applying to the course we must mention that you must be willing to work office hours everyday, from eight in the morning to five in the afternoon. We would also recommend you to not take any extra courses simultaneously since it will most likely make you fall behind in the project. Finally, we would like to stress the importance in that if you will take the course you must be willing and open to learn new programming languages and software.

2 Individual contribution to the project

2.1 Ludwing Franquiz

During the beginning of the project I was in the router group and started by investigating how the router cache works in CCN-lite alongside Maria. After finding out how it worked we changed the existing implementation so that it would fit our needs better.

I have also, together with Andrew, been responsible throughout the whole course for all the datastore related tasks and the connection to the GreenIoT project. First we put up a mongoDB datastore, secondly we developed an API so that the GreenIoT project would be able to query our data and thirdly we developed the MQTT gateway which creates a two way communication between our CCN network and Uppsala University's GreenIoT network.

Me and Andrew wrote the python program that retrieves data from the sensors and stores them into our CCN datastore. We also developed the named function networking part in our system which triggers different functions, such as querying historical data within an interval or executing the prediction function, depending on the prefix on the interests that the relay running on the same computer is receiving.

I have also been responsible for the report. Creating the structure on how it should look, proofreading (alongside Andrew), distributing the writing tasks and making sure that it is delivered on time.

During the end phase of the project we created a second Android application on which I wrote the functions for getting information from the directory service, retrieve sensor values and the refresh option (alongside Andrew and Max).

Finally I presented on both the mid-term and final presentation, I was the Scrum master holding the meetings and I was the bridge of communication between the students, staff and industrial partners.

2.2 Max Wijnbladh

Network
I was assigned the task of getting the CCN network up and running. After some research, we decided to use the CCN-Lite project as the base code and extend that system with the functionality we wanted. Among other things, I've developed the following functionality:

- Remove FIB entries from the relays using control messages
- Automaticly finding other relays running in the same LAN using broadcast messages and create faces towards them
- Auto populating the forwarding table with prefixes pointing at the face from which a named object arrived
- Registering/unregistering relays that boots up in the network to the DS
- Register faces that is created in the relay with the DS
- Update the FIB of a relay using a customized control messages sent from the DS and extending the existing control plane with the ability to create faces on a relay remotely.

Directionary Service
I was assigned the task of developing the Directionary Service (DS), which is responsible for managing the forwarding plane of our CCN network. From scratch I've developed a always-on python program that:

- Listens to incoming UDP messages and parses them based on type (JSON/CCN)
- Have the ability to add and remove sensors to/from the designated database
- Map the entire network (all the active relays, as well as the connections between them) and monitor it using UDP control messages
- Register the IP of the border-routers and calculate and update the FIB of each relay in the network with the shortest path towards those border-routers using UDP control messages

Android Relay
I was responsible for creating a customized relay that runs as a service on phones running the Android OS, as outlined in the project requirements.

This was done by customizing the existing android implementation of the relays provided by the CCN-Lite repo to fit our needs. For this part of the project, I've developed the following functionality:

- Coded an Android application that starts a service which runs a CCN relay, and bound the lifecycle of this relay instance to the service instance (since they run as separate processes)
- Implemented a "START ON BOOT" policy for the service, which allows it to start-up automatically when the phone is booted
- Added a persistant notification in the android phones notification bar, which allows the user to see which gateway IP the relay is connected to, as well as change the gateway during runtime, restart the relay, and monitor the statistics of that relay by pressing the notification.

Android Application
During the second to last week of the project, Me, Ludwing, Sami and Andrew decided that we needed an Android application that was more stable and bare-bone than the one that had been developed during the project. The goal was this application was to be used during the demo, as well as demonstrate that the project had fullfilled the outlined requirements. I collaborated with Andrew and Ludwing during the development of this application by coding and providing input on design choices.

Sequence Numbers
I came up with the technique we are using to calculate the sequence numbers of past content objects that has been produced by the sensors. This algorithms uses the start-up time/date and the lengths of the intervals that the sensor produces data to find the sequence number of the desired content.

Final Demo
For the final demo, I've collaborated with Sami to design and set up the architecture/topology of the network we're using. I've also written Python scripts that allows us to simply start, manage, or shut down our demo network. This includes scripts that are able to:

- Start up each relay in the network at the same time using remote SSH commands
- Create links (faces) between desired relays using a customized type of management command
- Shut down all relays or individual relays using a customized type of

management command

Other

In addition to the above contributions, I've also advised on various design choices in the project, developed various tools and techniques for testing and troubleshooting other parts of the project, contributed to the final report as well as the mid-term and final presentation.

2.3 Andrew Aziz

During the initial phase of the project, I worked alongside Max to create the base of the project which was the CCN router software. This software enables our CCN network to correctly propagate network packages from a node in the network to another. It also allows nodes to discover each other.

Furthermore I was also tasked with creating our database, a database API (alongside Ludwing) which allows different parties to query our database, the MQTT gateway which creates a two way communication between our CCN network and Uppsala University's GreenIoT network.

Me and Ludwing wrote the script that gathers data from the sensors and stores them in our database. We also wrote the script that triggers the named functions in our system depending on the prefix of the interests.

I was also involved in some minor roles, mostly fixing bugs and discussing plans and ideas, in the sensor group and the SDS group.

During the end of the project, we decided to create a second application for which I did most of the design and some functionalities.

2.4 Maria Rajabzadeh Namaghi

In the beginning of the project, I was in the router group and was dealing with the CCN-lite code to find some special parts for developing and also find how a router cache works in CCN-lite.

Then, I joined to the Android group. I created an interface for the team to test the CCN network. In addition, I tried to add some functionalities like having a slide menu and saving the results in a database. First, since we had an idea to show the locations of the sensors on a google map, I added a Google Map to the android application to show the hard coded sensor locations. However, this part was not applicable for the final idea of the application.

When we decided about the final android application, I designed the android application alongside Adrian.

Meanwhile, I implemented some methods for the prediction of values. One of them was in Java and a few of them were in R language. I investigated about ARIMA method and realized that it can be an option for our purpose and then I implemented that in R language. After a while, my teammate, Eric, implemented the same in python.

Furthermore, I implemented an interactive graph in the android application to show both the prediction and historical values.

2.5 Adrian Amigues

I have been the core member of the Android team from the start until the end of the project.

At the beginning my main task was to analyze and understand the ccn-lite library and especially the parts we would need to use for the Android application. From that I created some additional files that we would use and that we added to the library.

After that the android team and myself begun importing and modifying the existing ccn-lite android project to make it our own while still having a working testable application. During the span when the whole team had not yet decided on what the final application would look like or do, we begun creating a simple application that would test the ccn-lite network functions and get data from sensors.

Once the purpose of the final application was decided and the design mockedup, I begun its development. I did most of it myself and had a lot of help from Maria. This phase of the project required a lot of communication between the different groups because the communications went through all the parts of the network.

During all the development process of the android team I have been the git manager and I have helped everyone understand how to collaborate with it.

2.6 Petraki Eirini

In the beginning of the project, i joined the Android team. At this stage our goal was to import the ccn-lite android code, which was provided by SICS as an android studio project and built the basic application to test the relay. In this part of the project i learned Android programming and worked with JNI. That was a very beneficial learning experience for me and i think i have even further improved my Java knowledge through it.

During Sprint 2-3 Adrian, Maria and me made a presentation to suggest different versions for the application development. Furthermore, we performed a basic application testing with minimal functionality to test the CCN+Lite connectivity. Thereafter, i switched and joined the sensor group, where i initially worked on the sensor-border connectivity testing. We found the problem with the border-router connection, which was actually on the router and not our fault (bug) as we initially thought. To solve the problem we got a new border route.

Since there was no further need at sensor team me, Maria and Erik started to work at the prediction part. I studied different techniques for time series analysis and made a demo of the 4 most used method in R, with graphic visualization and also made a comparison of the different methods. We wrote some scripts to automatically request the prediction data as named functions.

2.7 Johan Snider

For the first month of the course I worked on the Android team. I helped with setting up the CCN-lite app to run in our development environment. Then I moved to the sensor team to help with border router problems they were trying to solve. I was on the sensor team for the last two weeks in October and then the border router problem was solved by getting a new border router.

After that I worked on a presentation of different naming scenarios we could use in the project depending on which type of functionality we were going to include. During this time I didn't really have a team. We were also discussing a lot about how to design and implement the SDS and also making the presentation for the mid-term presentation. Then on November 9th, we had our mid-way meeting at Ericsson and I was one of the people who presented.

After the mid-way presentation, I joined the SDS team. I wrote some small python programs to illustrate how I thought the SDS should be designed. But there were some disagreements about how the design should be and I found myself getting distracted by trying to make the predictions work with the data that we had collected. Then, I spent the last two weeks in November getting familiar with the IBM IoT platform so that we could use the default software on the sensors to collect data in Mobius. We were able to setup a sensor in Mobius for two days and this is the data that we used for the predictions part of the report.

For the remainder of the course, I've spent most of my time writing and proof reading for the report, which I think has been my most significant hard contribution to the project. I also spent time communicating with FooBar, Mobius, and UTN to see if we could install sensors in their lunch rooms which wound up not happening. I think in retrospect, I should have focused more on one part of the project instead of jumping around from topic to topic.

2.8 Theodosis Malatestas

I was part of the sensors group during the whole project. For the first weeks we were investigating the TI CC2650 functionalities. Since I did not have any experience on the sensors I was trying to figure out how the communication with the other parts of the system should be done.

Initially I tried RIME protocol and Serial Input events to try to debug the TI CC2650 and establish some kind of communication. After the arrival of the border routers we concentrated on using only 6lowPan all the way. Due to the problems with the first border router we were behind our sprint goals for about a month.

I spent around two weeks on making the microphone working but it was too difficult so I dropped it.

With the arrival of the new router the connectivity issues were solved, I joined the SDS group were we designed the final architecture. I made some small modifications on border router code, worked on modifying the sensor code to change the prefix dynamically and register the sensor on the requesting border router and redesigned content objects to fit the buffer sizes. I wanted to do some more improvements to increase the efficiency but it was too risky to implement these changes with a few weeks left.

During the last weeks I was helping with testing and writing the report.

2.9 Sami Kaivonen

I started this project in the router team. We began by investigating if we should use CCNx Distillery software or CCN-Lite software for creating the network for the project. After choosing CCN-Lite I was assigned to investigate how the chace worked on CCN-Lite platform and do we need to modify it. I was also heavily involved in design choices related to the network. Because of my lack of C coding skills I wasn't involved in the coding process but rather concentrated how to use the CCN-Lite software and how all the commands work. This led me to create a bash script for the CCN-Lite software that acted as a easy to use commandline user interface for the software to avoid human errors in the routing commands, that was very common, in our testing of the network.

After three weeks I was moved to the sensor team to help in their struggle with the STM32W border router. Thanks to wide testing and many attempts to solve the problem with the border router I became very familiarized with the device and the code and wrote step-by-step "how to use" -instructions, first for the sensor side of the project and later, for the sensor and router side of the project. Border router being a major challenge for our team the sensor team kept growing and I ended up instructing people how to set up the needed systems while trying to solve the problem at the same time. I also ended up helping people who had no previous experience with linux and had trouble using the Ubuntu OS. After dropping the STM32W border router and moving to the border router used in our final project I was moved to a demo team with Adrian for the mid-presentation. Unfortunately we couldn't show the demo in our mid-presentation at Ericsson because we forgot to bring the only Android device that had the latest application needed for the demo presentation.

After mid-presentation I was moved to SDS team. For me designing the SDS architecture was probably the most difficult, but at the same time the most fun, part of the project because most of the routing system is controlled by the SDS and all the functionalities needed to be take into an account while making decisions for the SDS architecture. I went through, with every team on the project, what was needed by the SDS and what would be required from the SDS by each part of the project. Then together with Max, we created the final architecture for the basic SDS that could be improved if necessary. We decided to create SDS with python scripts which enabled me to do my part of the coding. I created Register Service that is part of the SDS but runs on every machine that has a border router. Thanks to python I was also able to do minor improvements and troubleshooting on SDS and database scripts.

2.10 Eric Wang

I start working with Sensor team in the beginning. The TI Sensortag device we are using in the project is a new plattform for me so it took some time to get familiar with that and aslo we invegisted the functionalities by studying the offical example program. After that i succefully building communication between two sentag devices using UDP connection to transmitt sensor reading values so next task would be invoing border-router to connect our sensor network to the CCN relay. We were getting stuck for quite a long time with STM32 border-router and did many expertimets try to locate the problem. After several weeks of debugging we fixed the problem by getting a new Sparrow border-router from SICS.

The Android app design need to use the microphone sensor but there is no driver from Texas Instuments so the next task is to develope a microphone driver. From the datasheet we can see the sensortag has a microphone sensor and we found out that has a PDM output can be sampled with I2S module. But the task is more difficult than i thought. I tried to write the code by taking example from other sensors but have too many errors and recursive libiraries needed to be fixed. So we give that task up.

In the next stage i take the task Machine Learning for prediction as it's a functionlity the android app needed. After some theoritical study we decided to use ARIMA model as the sensor reading can be consider as time series. Then i did the implemention of prediction using Python.