Uppsala University
Department of Information Technology
Project CS 2010

OpenMSC
Course Report

SMS Capable GSM Network

Muhammad Yousaf
Umesh Paudyal
Tobias Vehkajarvi
Wenbin Wu
Tejas Oza
Papanash Muthuswamy
Premkumar Janagarajan
Johan Drake
Wenfei Zhu

Contents

Abstract

This document describes the project methodology called Scrum and the development process of the project which resulted in the SMS [1] capable OpenMSC , the major part of GSM[1] architecture.The project was developed by nine enthusiastic students of Uppsala University during autumn 2010.The goal of the project was to build a system capable of sending and receiving SMS following 3gpp specifications and GSM standards. As an end product we have OpenMSC developed in erlang/OTP [7] programming language.

Glossary

BTS	Base Transceiver Station
BSC	Base Station Controller
GSM	Global System For Mobile Communication
HLR	Home Location Register
MSC	Mobile Switching Centre
MS	Mobile Station
MAP	Mobile Application Part
SMSC	Short Message Service Centre
SIM	Subscriber Identity Module
VLR	Visitor Location Register
3GPP	3rd Generation Partnership Project

1 Introduction

The goal of our project is to implement an SMS[1] capable Mobile Switching Centre (MSC)[1] which is a part of the GSM[1] network that follows 3GPP standards. Mobile phones connected to our GSM network and registered in HLR[1] can send and receive SMS.

We would like to acknowledge Mobile Arts who provided us with BTS[1], HLR, SMSC[1] and the OpenBSC[9] which is an open source project of BSC[1].

The name of the project is OpenMSC since we are planning to make our project an open source. Our software along with OpenBSC can provide a foundation for testing and research purposes in GSM network. To our knowledge, there is no other open implementation of Erlang based MSC (Mobile Switching Centre) and we are the first to implement an MSC that is capable of sending and receiving SMS.

2 Project methodology and organization

2.1 Scrum

Scrum is an agile software development methodology that follows iterative model. It is fast, flexible and open to changes. It respects individuals and interactions between more than the tools and techniques used. It considers working software to be more important than the written documentation of the software.

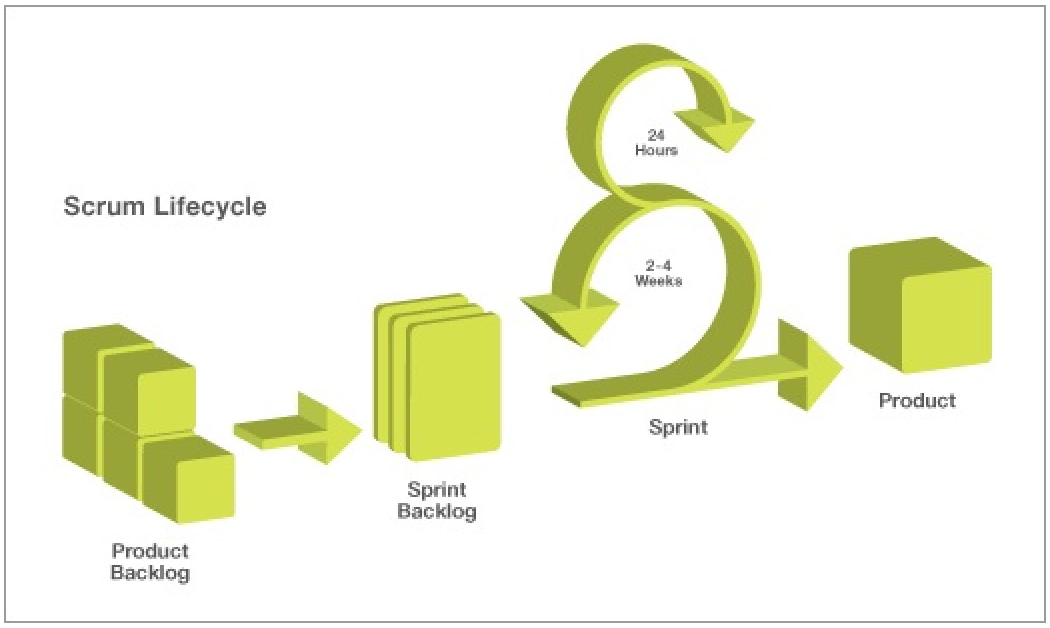

fig. Scrum Overview

Product Backlog The product owner creates the product backlog from the objective of the task. This backlog is flexible and is incrementally accomplished in the project. Tasks are prioritized as high priority and low priority tasks.

Sprint The project time period is divided into several sprints and each sprint usually spans between 2 to 4 weeks. The sprint begins with the sprint planning and ends with the sprint demonstration.

Sprint Backlog Sprint backlog is a subset of the product backlog that team is committed to finish on the bounded time-box. The backlogs are further split into smaller tasks.

Daily Meetings It is carried out to summarize what was done the previous day and what is to be done for the day.

Retrospective Meetings The sprint ends with sprint review and retrospective summarizing what went right and what went wrong and what can be improved in the next sprint.

Roles involved There is a Scrum Master who acts like an interface between the project owner and the group. He makes sure that the team follows the Scrum guidelines. He protects the group from external distractions. An ideal team usually comprises of 7 to 9 members.

2.2 How we used Scrum

2.2.1 Scrum Role

Product Owner Martin Kjellin and Jonas Falkevik the representative from Mobile Arts Company was the product owner of our project.

Scrum Master Scrum Master has lots of responsibilities. From our team except few people no one was interested to become a scrum master and we used to have some random selection to rotate the role as scrum master for each sprint.

Scrum Team In our project there was good mixture of people from different cultures and regions. But while working on project we worked as a whole team. Initially we started to divide the work in group of small sub teams according to our goals but later on we stuck to pair programming during the coding part.

2.2.2 Sprint Planning

Usually we used to keep the duration of one sprint as 2 weeks. Before starting the sprint we used to plan the goal that we had to accomplish at the end of the sprint. Sprint goals are the project backlogs that are divided in to small tasks. At the start of our project our goal was not that much specific as we havent got any Backlog from the company and we used to define backlogs ourselves but later on we learned many things from the project and we started defining specific goals and we got success in accomplishing them.

2.2.3 Daily Morning Meetings

Every morning we use to have a meeting among all the group member of the team. Every member had to explain what he/she have done yesterday and how and what he/she will do for the rest of the day. Including this the member also informs the group about what problem he or she encountered during the particular task so that the group members can discuss that problem and find out the solution. One of the benefit of daily morning meetings are everyone in the group are on the same page or every one knows what is going on in the project and recognizing earlier when problems need more manpower.

2.2.4 Sprint Demo

As our project was not backlog specific so after end of each sprint we used to present our achievement to our product owner. By doing this we keep our product owners in loop of our achievement and how we are doing.

2.2.5 Retrospective

At the end of every sprint we had a retrospective of that particular sprint. We all sit together and talk about what went good in the sprint, what didn't go well and what can be improved in the upcoming sprint. About what went good, we used to discuss on what made it to go good and try to repeat that in next sprint and about what didn't go good, we used to have a discussion why and what was the reason behind that and try to overcome it in next sprint.

2.3 Sprints

1st Sprint: 2010-09-16 to 2010-10-15

Scrum Master: Tobias Vehkajarvi During the first sprint we started by setting up our computers in our project room. We had already installed Ubuntu and Erlang on our computers in the two previous weeks during our introduction course in Erlang. We requested an additional workstation to use as a server in a project. Once we received it installed Ubuntu, Erlang, WM-Ware and GIT. We had some hardware problems with the server early on and had to change the graphics card.

Once we had all our resources set up we started reading documents. We read about SS7[5] and GSM in general early on. At first it was hard to find the documents that were most relevant for us i.e particular for SMS related and to know that they were relevant once we found them. Later on in the first sprint we found some reliable sources for documents and our understanding of GSM was now on a higher level helping us identify the relevant documents.

We also sent a request to 'Post och Telestyrelsen' in order to receive a spectrum licence for the BTS that we would use later on in the project.

2nd Sprint: 2010-10-18 to 2010-10-29

Scrum Master: Papanash Muthuswamy Once everyone in the group had a good basic idea of how GSM worked we started to read into GSM on a deeper level. We first tried to identify which parts of the MSC functionality we would need to implement in order to send and receive an sms. This was hard but we found some services such as Mobile Originating[2] and Mobile Terminating[2] which we quickly realised would be needed.

We searched for information about the protocols and messages to and from MSC that would be relevant but had some problems early on finding the information for the network layer that was most relevant. After some help from Mobile Arts we were able to find the correct documents.

The internal functionality of the MSC and VLR were our next interest. We started reading and looking at flowcharts of what exactly happened inside the MSC once it received a message from the outside. It was around this time that we started discussing the internal architecture that we would use in the MSC, a discussion that we would come back to again and again throughout the whole project.

Near the end of the second sprint, a few of the group members started researching the available OpenBSC versions. We found that there was a version of OpenBSC that had already implemented an interface towards MSC. We realized that if we used this version of BSC, we would not have to write or edit any code in the BSC to create our own interface, but instead only implement our own side of the already existing interface. At the end of the second sprint we were separated into two groups. One group researching and designing the architecture of MSC and the other group researching, installing and implementing the interface towards BSC.

3rd Sprint: 2010-11-01 to 2010-11-12

Scrum Master: Johan Drake We started finalizing a first draft for our architecture during sprint three. Ideas such as having temporary children perform all the work were already in the design by now. The interfaces that we would use to communicate towards HLR, SMSC and BSC started to be fleshed out.

Development of the Mobile Originating[2], Mobile Terminating[2] and Location Management[2] servers and children were started in this sprint. We only implemented very basic functionality and since the two interfaces towards BSC and HLR/SMSC were not finished yet we could not test our system as one unit.

CouchDB[6] was used to implement VLR during this sprint. We used a Erlang CouchDB to communicate with the database from our MSC server processes. CouchDB also included a web interface which was useful for manually checking and editing the information in the database while we were testing our servers.

Once we had researched more about BSC and SCCP[4] we realised we would either have to implement our own version of SCCP on the MSC side or use the library that was used in BSC. We chose to use the library that already existed. This forced us to write our side of the interface in C. We used Cnode for this. Cnode is written in C but for other Erlang processes it looks just like another Erlang process. This was useful for us since we could then both use the C library and communicate with our other Erlang processes in MSC using Erlang messages.

4th Sprint: 2010-11-15 to 2010-11-26

Scrum Master: Yousaf Muhammad The specification of the interface towards SS7[5] was finished in this sprint and provided to Mobile Arts. They used our specification to configure the SS7 stack for us so that we would be able to communicate with HLR and SMSC. In order for Mobile Arts to deliver their HLR and SMSC to us they needed to finish the implementation of their side of the interface we had specified and their configuration of SS7. Since we needed the HLR and SMSC as soon as possible to start testing our system, finishing our interface specification was our highest priority task in this sprint.

After learning more about OTP and supervision tree behaviours we started designing and implementing our own supervision tree. With some consultation from our friends at Mobile Arts we decided on a good design that enabled us to easily spawn and handle multiple identical child processes and keep each service separate from the others.

We turned our project into an Erlang application and tried to finish the implementation of all of our servers in this sprint. By the end of the sprint, we had all the minimal functionality for connecting a cellphone in the network, send and receive an sms. It worked in a testing environment but we had no SMSC, HLR, BTS or working connection to our BSC yet, so we could not test our MSC in a live network.

It was time to start using the SCCP protocol but confusion was now a fact. The two different type protocol classes, connection oriented and connectionless, was supposed to be used but none of us had the slightest idea of which BSSAP[3] packets belonged to which class. And the problem of how to actually implement a CNODE[7] to split and wrap the headers took us a while to grasp. Considering connection oriented messages we realized that references had a important part in the still unknown message flow and our services did not have support for this yet. But we started to build up a good conceptual model of which server was supposed to do what work. We introduced the parser server and decided that the children of the services would take care of the encoding by using the mbinterface.

A poster was needed at the Erlang User Conference to display the work that we had done. We started planning and designing the poster during this sprint.

5th Sprint: 2010-11-29 to 2010-12-10

Scrum Master: Premkumar Janagarajan MSC and BSC groups are merged. Receive HLR and SMSC and BTS.

The goal of the fifth sprint was to make sure that all the nodes of the system communicates with each other so that it is possible to send and receive messages.

We always had a problem of creating tasks which is specific and small since we didnt have a product backlog specified by the company. But during this sprint we managed to split the tasks into much smaller and more specific ones. That is because we have learned a lot in the previous sprints and also with the help of course assistants we managed to use scrum in a much efficient way during this sprint.

At the start of this sprint most of the modules were implemented and tested with the help of some simulated external nodes. Both the MSC and BSC groups were merged since the goal was to make sure all the nodes communicate with each other and the short message can be delivered from sender to receiver.

The company shipped the nanoBTS[8] hardware and some of the team members started to setup the hardware and the network. At the same time, the company installed SMSC and HLR and the whole GSM network was setup during the initial phase of the sprint. Since SMSC and HLR are external nodes we had a lot of problems communicating with them from/to MSC through the APIs. We had to make some kind of changes in the APIs which we provided to the company, for SMSC and HLR to communicate with MSC and vice versa. Since we only had the executables of the APIs the company provided to us, we spent a considerable amount of time figuring out the problems with the APIs.

We managed to test the MT[2] short message service using a SMPP client which ensured us that one part of the system worked. During the second week of the sprint we managed to integrate LM[2] service followed by MO[2] short message service. But still we had some bug fixes for services like ready for sms in LM and MO which we didnt manage to do in this sprint. But the basic goal of establishing a GSM network with our own hardware and software for sending and receiving short messages were achieved.

6th Sprint: 2010-12-13 to 2011-01-13

Scrum Master: Johan Drake In the last sprint before Christmas we finished the last bug fixes and added the last functionality needed to complete the project. We were able to connect our cellphones since last sprint. After some work with the interfaces we were also able to send and receive sms.

Those of us who were doing the presentation started preparing our slides on the last week before Christmas. We tried to distill the core parts of our project into a 30 minute presentation which was kind of hard. We realised that we would have to start with an introduction to GSM in order for anything to make sense. We wanted to show a detailed picture of our architecture and message flow where a large part of our work had been spent. We decided to show a part of the message flow in MO as an example since it was very similar to the rest of our system and was short enough to explain extensively in a presentation.

The group members who did not work on the presentation, instead started working on our Project and Course reports. We wanted to get started with them and have a good idea of what was left to add to them by the time we got back from Christmas. We wrote the reports in google documents, then transfered the text and formated it in LaTeX.

After Christmas we have used all our time for discussing the contents of the reports and writing the reports.

3 Resources

3.1 Man Power

Nine enthusiastic students at Uppsala University autumn 2010 with different backgrounds worked together for the same goal SMS capable GSM Network. Some of us had some knowledge of GSM where as most of us had to start to learn the GSM specifications. But it was pretty interesting and challenging at the end. The group constellation included two swedes, three indians, two chinese, one pakistani and one nepalese. So, it was good experience working with people from different cultural backgrounds. All of the group members were some what new to erlang but with our functional programming background that helped us getting started.

3.2 Hardware

All of the group members were provided with one Quad Core HP computer each and one extra computer was dedicated as the project server. Ubuntu was the obvious choice of operating system for all of the computers since we wanted a common platform to make sure that everyone was using the same versions of the involved software. The server served us many different services such as running RedMine for managing scrum, git repository for our version control system and two virtual servers running a SMSC and a HLR from MobileArts. MobileArts also provided us with a nanoBTS as our BTS but the only problem was that it came without manual and that kind of information is not freely available. But fortunately this was quickly sorted out with the help of kind hackers from OpenBSC that had reverse engineered configuration software and some pointers on how it works. For testing our GSM network we used our own cellphones to connect and for generating SMS traffic.

3.3 Software

In the start of the project we tried to use RedMine as our scrum organizing tool but it gradually became more and more unused since scrum did not fit our project in that sense and a simple pin-board was enough. Emacs, VIM and eclipse were the most prominent developing platforms used and they all worked well with our choice of version control software that was git except that none of us had used it before so it took some time to get acquainted with it. For traffic flow analysis and debugging purposes Wireshark was our most important tool. We used OpenBSC which is a open source implementation of a BSC and has support to be used as a stand alone component in comparison with the default version that implements a complete but minimal version of a GSM network. For our database solution in VLR we decided to use CouchDB and it suited us very well since it was easy to setup and we really did not have that many requirements on performance.

3.4 Literature

We were offered to buy the book Erlang programming by Francesco Cesarini and Simon Thompson at a reduced price in the beginning of the course and several group members took the opportunity and it was a great erlang resource. 3GPP specifications were freely available on the 3GPP homepage but it was a real challenge to find. And knowing which of all these hundreds of documents were the correct ones to read was a minor project of its own because they often were nearly thousand pages long with poor descriptions of contents.

3.5 Environment

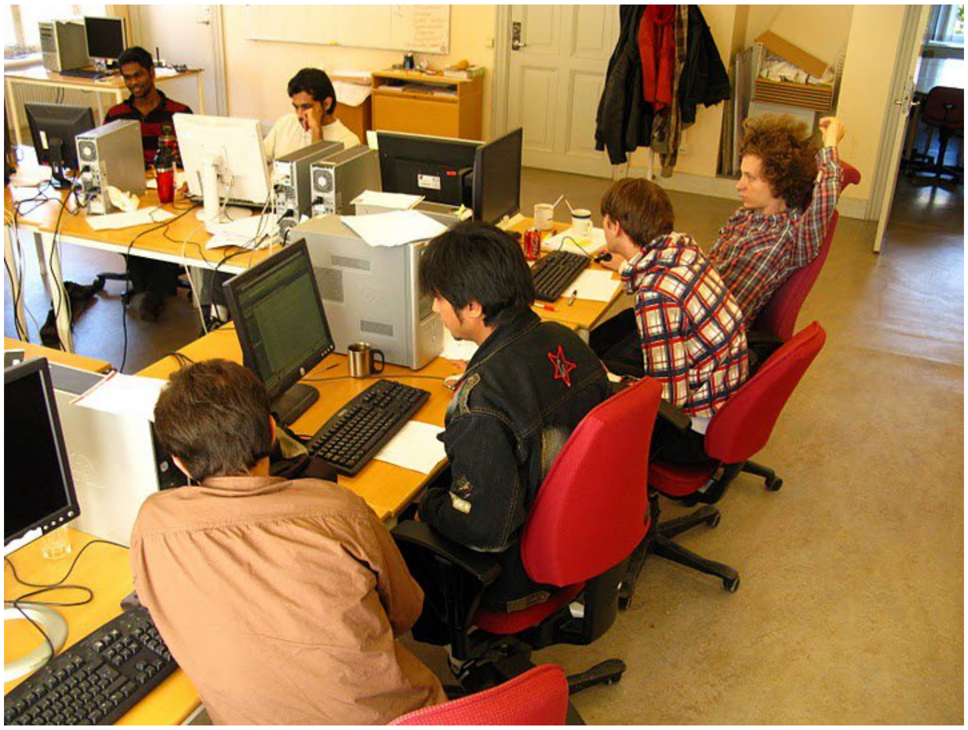

fig. Project Room

The above picture is the place of our project room where we have been working for 4 months. Project room is located in Hus 4 of ITC building at 4th floor, which stores the view of nice surroundings. The project room is large enough to accommodate at least ten people with computers where two teams were sharing same common kitchen with one common entrance to each other room. The kitchen has a nice table where ten or more people can have lunch at same time and also it has small white board, one refrigerator, two microwaves. We found the kitchen as complete place for sprint planning, retrospective, meetings and coffee breaks. When we had chosen the project we were told to arrange the tables so everyone could work comfortably. We arranged the tables in U shape where everyone can see and communicate to every other and this arrangement was just perfect for 9 people.

3.6 The Company

Mobile Arts was a real good source of information since they are working in the field of GSM networks so whenever we ran in to difficulties about the architecture they would quickly help us out and get us back on track.

3.7 The Community

The friendly people on the OpenBSC mailing list really gave us good pointers and also their official irc channel #openbsc on freenode was a great resource. For erlang problems you could always consult the irc users at the #erlang channel on freenode for almost instantaneous answers.

4 Problems

Problems with Scrum For a number of reasons, the Scrum methodology was hard to apply to our project. In scrum you usually want to start out with a number of clear backlogs that can later be divided into tasks which are given time estimates. Splitting backlogs and giving time estimates is hard when no one in the group is familiar with the material we are working with. It is hard to know where best to start without first doing research. During the first month of the project we were mostly doing research and throughout the project, research was a large part of the project. Even the task of researching is however hard to shape into a scrum friendly task especially since it is unclear what is important to read and how long it will take before you have started.

In scrum, a task should preferably be small enough that it can be finished in one work day. We had problems creating tasks that were small enough. We were often only able to split our tasks down to a certain level, both because of our lack of knowledge about what the task would entail, but also because of the nature of our tasks. Sometimes we had to scrap tasks during the sprint because we had made assumptions during the sprint planning that were unfounded. Other times we would create huge tasks during the sprint planning that would later, during the sprint, be split into smaller tasks, once we knew more about them.

There was never really a list of backlogs that we could use as a source for new tasks during our sprint plannings. We usually just decided on a goal for the sprint and from there, created new tasks from scratch. It was hard for us to maintain a list user stories other than the one we started out with, Send and receive an SMS since we were not completely sure what that user story would require us to implement until half the project was done. Often we could not predict what functionality we would need to implement more than one sprint ahead of us, sometimes not even that.

In scrum you want to have a simple running system up as soon as possible. More functionality can then be added in each sprint. For many reasons, this was hard for us to accomplish. One reason for this was that we were creating a very complex system from scratch. The most simple version of our system was already rather complex. We also needed to communicate with other nodes that required us to use the protocols they had defined. We were also unable to actually test our code against these other nodes early, since we received them in the fifth sprint. Before that we could only simulate them. We did however manage to get all our nodes to at least perform some basic communication quite fast after we received them.

Unproficiency in Erlang It took some time for all of us to get acquainted with the coding in Erlang language as most of us were totally new to Erlang language. We worked for architecture design most of the time due to the lack of knowledge of design principles specific for erlang;like supervision trees forced us to do some redesign later in the project.

Communication between the two groups We happened to divide the two groups, BSC and MSC group, to make the concurrent development on two separate modules but unintentionally it turned out to look like two separate groups without much communication. Then we had to reshuffle the group to improve the communication in between the groups. We also tried to introduce a weekly internal demo to increase the insight of the both groups progress and making sure we were on the right track with our design.

Lacking documentation Since we were using OpenBSC as a open source implementation of BSC written in C we had hard time to understand the implementation as it did not have any proper documentation to follow. We had to face the same problem while configuring the nanoBTS as we did not get any technical/installation manual and none was available for free except for some reverse engineered information.

Keeping logs We did not keep a good record of our sprint goals, retrospective conclusions and exactly what we had worked on in each sprint. We had some information written down but when writing this report we had to do some detective work to figure out exactly when we did everything. If we had kept better personal logs and more structured protocols of all our scum meetings, writing this report would have been a lot easier.

Tip for future student's While writing report we realized very late to seperate Design and Implementation chapter in Product Report and this chapter was tightly coupled and due to lack of time we were not able to do that. This is an nice advice for future student to emphasize on this topic while writing Product Report and to write seperately design and implementation part.

5 Conclusion

Our project goal was to make SMS capable GSM network. So, accordingly we have been able to send the SMS using GSM Network successfully. In our four months of project we believe that it was one of the best course we have had in our master degree. It is recommended to every master student that they should take this course because this course gives you advantages from every aspects like research, programming, team work and more importantly interaction with leading companies in the field of computer science. It was fun to work with students having different background and to know each other. We were not able to implement scrum effectively in our project because of research based nature of the project but we tried to implement as much as we can. For each of us, working with telecom industries and implementing it in erlang has been a good and handy experience.

6 Individual Contributions

6.1 Tobias Vehkajarvi

In the first half of the project I was more involved in the MSC Group as we called it. The first couple of weeks were dedicated for reading and understanding the GSM network architecture in general, setting up the server with some of the software and I was also the scrum master for this sprint. Johan and me filled in the forms for the BTS spectrum license and sent it too PTS. I was involved in the design of our initial system design. When it was time to work on the actual services in MSC I started doing flow chart analysis of location management service. I was also in charge of making the arrangements for our Kick off with the company, it ended up being laser game and good food and drinks at the African restaurant called Messob in Uppsala. When most of the communications had been identified for the services I was involved in doing the first draft of the SS7 interface and started working in the other group called BSC Group. Started implementing the library that was needed for creating and parsing packets for BSSAP protocol which I worked a lot with through out the rest of the course. As a result of this I was also closely connected to the development of the interface between MSC and BSC the so-called mbinterface. I was helping the people coding the CNODE in how SCCP actually works with connection less and connection oriented message, it took a while but we finally got it. The work done during this period was done without having any real traffic going on with the BSC so focus was on writing documentation, specifications and common test code for the BSSAP library functions. Attended the erlang user conference in Stockholm and made several useful and interesting contacts with people from the industry that will surely help me in the future. In the end of the project I did code refactoring for the supervisions tree by developing a more general module that replaced most of the old supervisors and also extensive code documentation. I spent time on developing a more robust and flexible version of the BSSAP library that almost went live but due to time restraints it did not.

6.2 Tejas Oza

When the project was started in September we learned the basics of Erlang and I started with finishing my Erlang exercises soon after when we have provided the SS7 documentation and link for 3gpp I have done lot of reading in SS7. The 3gpp website is very big and contains hundreds of documentation so choosing the correct one was also a challenge. After we got the correct documentation First 2 weeks I was reading the MAP specification and found that all 9 members were doing the same thing and lots of time was being wasted so I decided to split into two sub groups i.e BSC and MSC group. I decided then work with BSC and started understanding the OpenBSC code that was written in C. I also joined the OpenBSC mailing list as there were no documentation. I got the hint that we need to develop the CNODE that splits and wraps the network packets, So I developed the simple CNODE to understand the concept. After talk with Holger Hans Peter Freyther, the author of OpenBSC code, I did lots of reading on SCCP and BSSAP and started developing CNODE with SCCP functionality and was inovlved in building the full functional CNODE. In Parallel I was contributing in implementing the encoding and decoding i.e splitting and wrapping of network packets including BSSAP and SCCP protocols. Later on I was involved in developing mbinterface. After we got the nanoBTS I started figuring out the synchronization OpenBSC and nanoBTS and make changes to configuration file so that OpenBSC can talk with our OpenMSC. On later part I was involved in fixing bugs in mbinterface, CNODE, encoding and decoding. When we had a presentation in MobileArts myself and umesh prepared the slides.

6.3 Johan Drake

During the first week of the project Tobias and I requested and installed a server with Git and other useful software for our development. This server would later also be used by the company to install their SMSC and HLR. Tobias and I also researched how to send a spectrum license to PTS for our BTS. During our project we had four presentations. The first presentation was an introduction to the project for the first year students. This presentation was prepared and presented by me and Premkumar. The second presentation was presented by me and Premkumar when we visited Mobile Arts. We had one course presentation before Christmas and one in January. The first was prepared and presented by me and Yousaf. The second was presented by me and Papanash.

I was involved in designing the architecture of our system and defining the interface towards SS7. Once we were developing our system I wrote and designed most of our supervision tree. I later converted our project to an Erlang application. We also needed a log server in our project which I designed and implemented.

I helped design the poster which we used at the Erlang User Conference.

6.4 Wenbin Wu

In the first 2 months, I read a lot about GSM network. Besides, I designed the system architecture with others. And then, I joined in the BSC team since they needed help. In the following several weeks, I read the code of OpenBSC and started to write CNODE. It took me quite some time. First, there is no document in OpenBSC, I need to read all the code to understand how it works. Second, CNODE takes charge two connections in two directions which means there are two processes in CNODE. I spent almost one week doing this and making sure the traffic is right. Then I worked in parsing and wrapping BSSAP packets with Tobias. At the same time, I fixed small bugs in CNODE. After getting nanoBTS, I tested the whole system and fixed a lot of bugs.

During the project, I installed RedMine and git in our server. And I taught others how to test code using Common Test and Git. I also contributed patches to OpenBSC.

6.5 Wenfei Zhu

During the first couple of weeks, I was reading and understanding the GSM network, the mapp protocol. I was involved in designing the architecture and implemented a demo of that. I implemented a supervisor to supervise different server, a msc_listener to dispatch different messages to corresponding servers, and an mo_server which will spawn a mo_child when it receives an mo_forward message. Then mo_child was a gen_fsm.

Later, I took part in defining the interface between MSC and SS7 stack. I implemented the mapp node and deployed it on the server finally.

In the sprint planning, I was assigned vlr implementation, and started looking for a database and erlang interface. I installed CouchDB and the interface, erlang_couchdb. After that, I was learning about CouchDB and its view concept, and figuring out how to use erlang_couchdb. I studied the flow chart of vlr for MT and MO module and implemented them. I also studied the messages from vlr to hlr, and implemented them.

Apart from that, I am also responsible for MT module. I learnt the flow chart and messages should be used in mt, and wrote mt_server and mt_child according to the flow chart. I discussed with people from BSC Group about the interface for messages sending and receiving from the MS.

During the last sprint, I did test on MT and VLR, and documented my code.

6.6 Muhammad Yousaf

In the beginning of the project, since we are beginners to GSM so i have to do lot of study and research about the GSM architecture, design, protocols, interfaces and 3gpp specification. It took lot of time but was worth it.

Then I joined BSC project whose goal is to understand the implementation of OpenBSC. It was written in C language and big problem was no documentation so we has pretty tough time in getting the grip on the implementation of OpenBSc. To make use of OpenBSC we had to implement a node known as CNODE that can communicate with OpenBSC. I implemented the splitting and Wrapping of SCCP packets in CNODE. I was also involved in communication between different components of network.

I was also interviewed by one groups of rst-year students for their assignment concerning our project. I was also involved in implementation of parser whose task is to parse the packet according to the protocol that messages accept and forward to others components of the network.So I was involved in parsing packets like SCCP, BSSMAP, IPA, DTAP, DT1 and UDT. I was also involved in implementation of mbinterface.

I was also scrum master of fourth sprint, In this sprint along with project goals we had an intresting conference known as erlang user conference. I built a poster for erlang user conference for presenting our project to the industry. Erlang user conference was very helpful in getting PR from the industry, that I think will be very helpful for me in the future. During the project we had four presentations and I was envloved in presenting one of them with Johan Drake. In the end I was involved in documentation of product and course report and converting in LATEX.

6.7 Umesh Paudyal

Initially, I started with lots of reading on GSM architecture and 3gpp specification as I was totally new to the telecom technology. Also, I had to spend lots of time in reading and understanding of open source implementation of BSC due to lack of any proper documentation and same happened while configuring the BTS as it also lacked any documentation. Later, I was involved in system architecture design for communication between OpenBSC and MSC which typically required tasks like understanding the OpenBSC implementation written in C and to make packets like SCCP, BSSMAP, IPA, DTAP, DT1, UDT to ensure the communication between OpenBSC and our erlang implementation of MSC. I started with the CNODE implementation to communicate between OpenBSC and MSC and continued to work on the so called BSC group.

In the midway through the project, we had to attend the erlang conference where I got the opportunity to meet different people from different companies contributing in erlang community. I along with rest of the members presented what we were doing in poster form in the same conference. Similarly, as a part of course assignment, I was interviewed by two first year student groups.

Later I involved in the group named as MSC group working on other implementations like Location Management services. In the meantime, I had to develop the interface named mbinterface to enable the communication between different servers like Location Management, Mobile Originating and Mobile terminating and the CNODE. In the final weeks of the project, I contributed in the development of parser for different packets along with testing of our implementation with the hardware called nano BTS and debugging as well. By the end of the project, I was involved in testing of the system along with most of the report writing in Product report and Course report and converting the reports in LATEX.

In summary, despite having other programming language experience, the learning experience of functional programming language like Erlang has been exciting through out the project along with the exposure to the software companies working in telecom industries became handy in gaining handful of knowledge on one of the telecom technologies like GSM. Following an agile project development methodologies SCRUM unlike the other methodology I was used to, added the experience on knowing how working in sprints with smaller tasks breakdown(i.e. backlog) becomes handy in software development. Working in a group of different cultural backgrounds has also been a good experience for me.

6.8 Papanash Muthuswamy

Since the beginning of the project I was with the MSC group which took care of the implementation of the MSC. In the first sprint Ii spent most of the time reading about SS7, GSM architecture and the protocols from the 3GPP specifications. During the second sprint we started to design the architecture of our MSC and I was a part of it. The design decision took a lot of time where I did some research about Erlang/OTP and supervision trees. I was the Scrum master for this sprint and I was also Involved in specifying the interface between MSC and SS7.

During the third sprint, I spent time understanding about the location management and Identity management procedures. I also had to read about the SMS management and all other related stuff to get a complete understating of the procedures. During the fourth and fifth sprints I implemented the LM module that includes location updating, ready for SMS, identity request and tmsi reallocation procedures. I also implemented some VLR functions that are needed for the LM module. Since the module communicates with HLR, I spent some time understanding the HLR that was provided to us by Mobile Arts.

In the later sprints,Ii was involved in testing the system. I spent a considerable amount of time in testing since there were several cases in the work flow of L2 functionality. I tested many errors conditions that can arise from both HLR and BSC. They need to be properly handled in the MSC. During the last 2 weeks I spent most of the time writing reports for the course report and product report. Also, I did the final presentation with Johan.

6.9 Premkumar Janagarajan

Just like others, I spent a whole month reading about GSM network and protocols trying to figure out where and how to start. When everyone had a basic idea of how GSM works and the protocols involved I was involved along with others in the system design. Since none of us were proficient in Erlang, we spent much time discussing how to design the system using OTP design principles.

At this time I was assigned with the task of understanding the MO short message service part which naturally made me a part of MSC group. I started with that and finalised the work flow of MO. I started implementing a prototype of the MO module. At this time prototype of different modules were implemented. Once we had a prototype of the design and some code, the company people took a look at it and gave some valuable suggestions on things like how Erlang supervisors can be used for our system.

Then I started reading about Erlang supervisors and did a research on how and what kind of supervisors can be used for MO service. I took the suggestion from the company of having a child supervisor and a server under a main supervisor and implemented it for the MO module. Later, all the work under supervision tree was carried out by Johan.

After the primitive implementation of MO was done, we needed SMSC to test it. So I have simulated a simple SMSC for testing and for showing demos. Also involved in understanding the work flow of MT since we havent started on it for a long time. So I and some of us were involved in finalizing the work flow of MT which was implemented by Wenfei in a really short time.

Once we got the real hardware and the SMSC and HLR, I had to again work on MO to make sure it works with the other nodes. So I was also involved in some of the parts in the APIs which MO uses to communicate with SMSC. Before doing this I was assigned to understand the new software installed in our server, the SMSC and HLR. I and Yousaf spent quite a lot of time exploring how the newly installed software worked. Then I finished the implementation of MO and worked with the BSC group to achieve the full flow of it. Also managed to communicate and deliver the short message to the SMSC.

I also served as a communication channel between the company, the university and the team to streamline the communication. I was the scrum master for the fifth sprint. I and Johan have presented the Project Course for the first year students during the start of the course. I have also presented the project along with Johan to the employees of Mobile Arts company during our visit to the company in Stockholm.

7 References

1. Mobile Communication (2nd Edition), Jochen Schiller, 2003

2. 3GPP GSM 09.02, Mobile application part specification, 1998

3. 3GPP GSM 08.08, MSC - BSS interface layer 3 specification, 1999

4. ITU-T Q.712, SCCP

5. TietoEnator Internal SS7 course material.

6. CouchDB: The Definitive Guide by J. Chris Anderson, Jan Lehnardt, Noah SlaterPublisher:O'Reilly Media, Released: January 2010

7. Erlang Programming, A concurrent approach to software development, Francesco Cesarini and Simon Thompson, OReilly, 2009

8. www.ipaccess.com/en/nanoGSM-picocell
(Referred on November 2009)

9. http://openbsc.osmocom.org/trac/
(Referred on October 2009)