MoPeDi Project

Project Report

Olli Alanen, Kimmo Haukimäki, Timo Juonoja, Petri Rönkkö

Version 1.0, January 20, 2004

University of Jyväskylä
Department of Mathematical Information Technology

Abstract

MoPeDi project developed a generic middleware component that is used between Bluetooth's transmission protocols and different kind of mobile peer-to-peer applications. The product is called BlueCheese. The test application in the project was a dummy application with needed features. The project report describes how the project was carried out and compares the realized project to the project plan.

Contents

1 Introduction

MoPeDi was a student software project in the Department of Mathematical Information Technology at the University of Jyväskylä. The customers of the project were Agora Center and Mobile Computing line in the Department of Mathematical Information Technology at the University of Jyväskylä.

During autumn 2003 the project implemented a generic middleware component called BlueCheese. BlueCheese is used between different kind of mobile peer-to-peer applications and different kind of transmission medias.

BlueCheese runs in mobile phones, but it doesn't use the GSM technology in the basic data transmissions. In the project, Bluetooth was used as a transmission media. Another possible choice could have been Wireless LAN, but because it is not yet available in today's smartphones unlike Bluetooth, the software does not support WLAN yet.

The aim of BlueCheese is to provide interface between the user interface and the transmission protocols. In Bluetooth's case the transmission protocols are L2CAP and RFCOMM. BlueCheese was implemented to Symbian OS using C++ programming language. The test application for BlueCheese was a dummy application with needed features.

In addition to this document separate documents have been written for the specification and the software design. The test report was also written and it includes the test cases that were carried out and their results.

The structure of the project report is as follows. Chapter 2 describes used terms and abbreviations. Chapter 3 presents the backgrounds behind the project while Chapter 4 introduces briefly the software. Chapter 5 describes the resources used in the project and Chapter 6 shows the realization of the tasks. Chapter 7 presents the reached goals and Chapter 8 the realized schedule. The realization of the risks is discussed in Chapter 9 and Chapter 10 explains the experiences of the project members.

2 Terms

The list below explains the terms related to the project.

Bluetooth is a global de facto standard for wireless connectivity. The technology is based on a low-cost, short-range radio link that operates in a globally available ISM band at 2.4 GHz, making Bluetooth usable worldwide [9].

GPCS (Gasoline Price Comparison System) is a mobile peer-to-peer application, which can be run in mobile devices.

GSM is an abbreviation from Global System for Mobile communications.

ISM refers to radio frequencies in the Industrial, Scientific and Medical band allocated at 2,4 GHz. This ISM band is unregulated, therefore no license is needed to operate in it.

L2CAP is a layer of Symbian OS's Bluetooth protocol module. It is located below the RFCOMM and the SDP protocols. L2CAP's tasks are the segmentation, the reassembling and the protocol muxing.

Middleware is a software that provides a programming model above the basic building blocks of processes and message passing [3].

Peer-to-peer (P2P) is a computing concept which enables sharing of the computer resources and the devices by direct exchange. This allows equal peers on the edges of a peer-to-peer network to connect and share resources without a centralized server.

RFCOMM is a protocol located on the top of the L2CAP protocol. It emulates the RS232 serial port and in this way offers an API to software developers.

SDP is a protocol located on the top of the L2CAP protocol. It handles the service discovery of the Bluetooth devices.

Symbian OS is an advanced open standard operating system for data enabled mobile phones. It includes a multi-tasking multithreaded core, a user interface framework, enablers for data services, application engines and wireless communications.

WLAN (Wireless Local-Area Network) is a type of local-area network that uses high-frequency radio waves rather than wires to communicate between the nodes.

3 Backgrounds

The chapter describes the backgrounds of the project. The organizations behind the project and reasons for starting the project are also explained.

3.1 Organizations Involved

There were two organizations behind the project: Agora Center and Mobile Computing line in the Department of Mathematical Information Technology at the University of Jyväskylä.

In Agora Center, there is a project called InBCT. The project works on the areas of business, communication and technology. A primary objective of the project is to create new and to develop existing know-how in the area and to facilitate the transfer of this know-how to the operating companies. Cheese Factory is part of the InBCT and concentrates to peer-to-peer networks. MoPeDi project was a part of Cheese Factory.

Some companies including Nokia had been interested in the practical mobile peer-to-peer applications. Therefore the needs of the companies were also one motivation for the project.

3.2 Gasoline Price Comparison System

The idea for the Gasoline Price Comparison System (GPCS) application was originated by professor Jarkko Vuori. GPCS is a mobile application, which can be run in mobile devices. It's task is to collect information about the gas prices of the different gas stations and then make decisions on where to refuel.

In the system a mobile device gets the information of the gas price from the gas station where the car is refuelled. Then it starts to spread the information to other mobile devices it encounters as two cars come across. Technical supervisor Oleksiy Volovikov is doing his master's thesis on GPCS and he is developing an application for that.

Graduate student Matthew Kam from the University of California Berkeley visited University of Jyväskylä last summer and he interested in the idea of professor Vuori. He made two slide shows [7] and [8] about the topic and started to develop a simulation environment for this kind of application. He is going to simulate different kind of areas with different car counts. This way he will find out how many cars are needed to spread information enough and how fast does the information spread in a mobile peer-to-peer community. The simulation studies were supposed to be concluded during autumn 2003.

4 Introduction to the Software

The software developed in the project is named BlueCheese. It is a middleware component for Symbian OS smart phones. It uses Bluetooth as its transmission media, but other medias like WLAN could be used in future. BlueCheese provides its services to multiple applications at the same time, but it can be connected to only one other device at the same time.

BlueCheese automatically finds new devices and establishes connections to them if they haven't recently met. It also disconnects automatically after all the applications have exchanged their data. BlueCheese provides also a locator service that gets current location from the GSM base stations and compares locations to the current one when needed.

BlueCheese handles queuing of the packets in the receiving and in the sending side. It also queues the other devices when it has a connection established with another device.

BlueCheese was planned to be tested with a Gasoline Price Comparison System application. The application however was not ready enough early and the testing was done with a dummy console test application. The features of the test application are described in Testing Report.

The software was developed under a public license. The license used was the Academic Free License 2.0 [1].

5 Resources

MoPeDi project used human and material resources in order to be completed.

5.1 Human Resources

MoPeDi project consisted of the following four students studying at the University of Jyväskylä:

- Olli Alanen (`opalanen@cc.jyu.fi`),
- Kimmo Haukimäki (`kikahauk@cc.jyu.fi`),
- Timo Juonoja (`tijuonoj@cc.jyu.fi`) and
- Petri Rönkkö (`pronkko@cc.jyu.fi`).

The project members Timo Juonoja and Petri Rönkkö are majoring in the Telecommunications line. Olli Alanen is majoring in the Embedded Systems line and Kimmo Haukimäki is majoring in the Software Engineering line.

The representatives of the customer were Matthieu Weber from the Department of Mathematical Information Technology and Mikko Vapa from the Agora Center (InBCT/Cheese Factory Project). The supervisor in charge was Jukka-Pekka Santanen from the Department of Mathematical Information Technology. Oleksiy Volovikov and Niko Kotilainen worked as the technical supervisors of the project.

5.2 Material Resources

The department of Mathematical Information Technology provided a project room with five workstations, phone and office supplies.

One of the workstations in the project room was running Linux Red Hat Release 9.0 and four of them were running Windows XP. The implementation tools of the project were Series 60 SDK 6.1 for Symbian OS and Microsoft Visual Studio 6.0. The version of Series 60 SDK for Symbian OS is compatible with the following mobile devices: Nokia 7650, Nokia 3650 and Nokia N-Gage. The project used one Nokia 7650 and one Nokia 3650 mobile phone for testing. The programming was carried out with C++ and the documentation was written using LaTeX.

The project room C225.3 located in Agora and it's phone number is 014-2604971. The E-mail list `mopedi@korppi.jyu.fi` was created as a communication channel of the project. The list archive can be found from `http://korppi.jyu.fi/list-archive/mopedi/`. The homepages of the MoPeDi project are located at `http://kotka.it.jyu.fi/mopedi/`.

6 The Tasks and Their Division

The chapter describes the tasks of the project group and their division to the members. The workloads of the project are also

6.1 The Workloads

The planned tasks for each group member can be found from the project plan. Some of the tasks were modified during the project. The realized task division of the whole group can be seen from Table 1. The table also contains the planned and the realized workloads.

Task	**Realized (h)**	**Planned (h)**
Meetings	120	-
Lectures	71	-
Presentations	44	-
Meeting reports	43	40
Project plan	50	100
Specification	150	150
Software Design	152	200
Implementation and testing	490	470
Test report	10	90
Software report	60	80
Project report	20	60
Manual	10	8
Homepages	10	10
Project Management	20	25
Unclassified	150	-
Total	**1400**	**1233**

Table 1: The realized tasks and their workloads.

As it can be seen from Table 1 the planned and the realized tasks match quite well. The total amount of the workload is bigger than planned and that can be explained with some tasks that we didn't consider at all in the plans.

6.2 The Changes

The implementation and the testing phases were merged to one task, because they had to be done side by side during the whole time. The testing's proportion of the implementation phase was smaller than we planned, but the whole time usage of the phase was very close to the planned one.

The task division was planned pretty well and a lot of changes didn't have to be done. The proportion of the unclassified tasks seems to be quite big, but that's because some times in the time usage reports weren't written perfectly. Other tasks took pretty much the same time that they were planned.

6.3 The Tasks of the Project Manager

The next tasks were the most important duties of the project manager:

- the project scheduling,
- informing the project members, supervisors and customers on the decisions, tasks and problems related to the project,
- reserving a room for the meetings and
- motivating the project members.

Kimmo started as the project manager but in the middle of the project Olli was changed to the project manager. The reason for this was that Kimmo's responsibilities related to the implementation seemed to take very much of his time.

6.4 The Tasks of the Implementation

The implementation phase was divided to the tasks described in Table 2. Kimmo's and Timo's tasks proved to be the largest ones. Especially the Bluetooth interface included a lot of classes and so the other members also participated on the implementation of it. The classes `ReceiveQueue` and `BcCore` were moved to Olli as a result of the same reason. Another reason was, that `BcCore` and `NeighborDatabase` were much more easier to implement than we expected.

Task	Person in Charge
`BcCore`	Olli
`ReceiveQueue`	Olli
Bluetooth interface	Timo
Location service	Petri
Application interface	Kimmo
Neighbor database	Olli

Table 2: The realized implementation tasks.

6.5 Workloads in the Task Types

The distributions on the time spent on each task type for each project member are shown in Figures 1 - 4.

As it can be seen from Figures 1 - 4, the task division was quite equally distributed. All the members participated on every task, but the charging person on each task made a bit bigger part than the others. All the members also had almost the same overall time usages (about 330 hours). The more detailed time usages can be found from the project's folder.

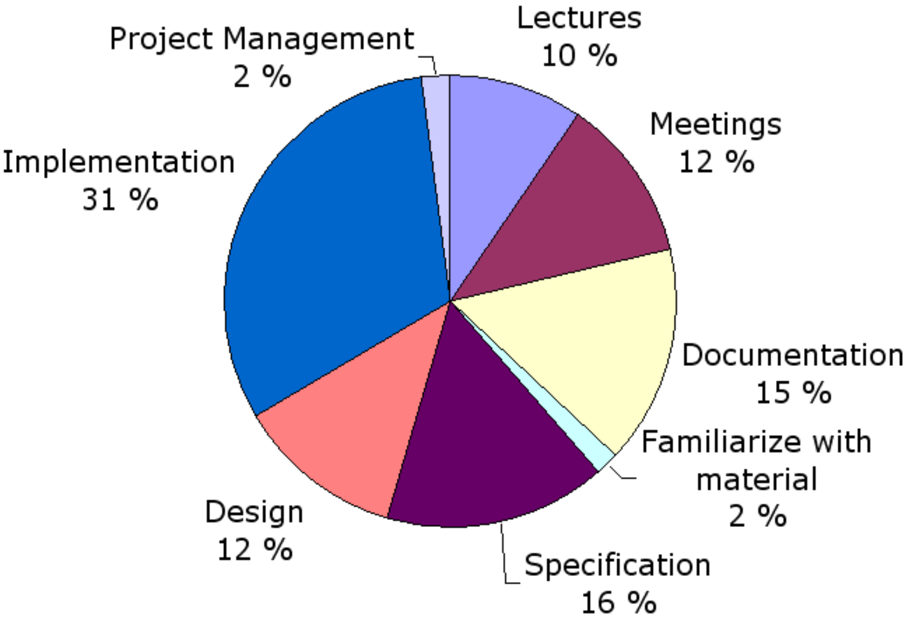


Figure 1: Olli's workload in tasktypes.

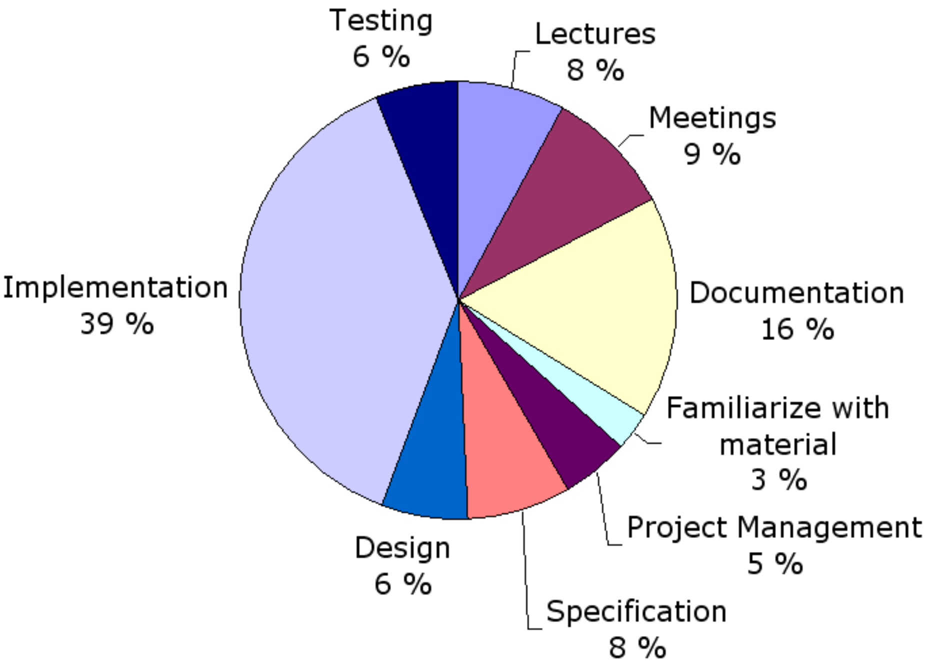


Figure 2: Kimmo's workload in tasktypes.

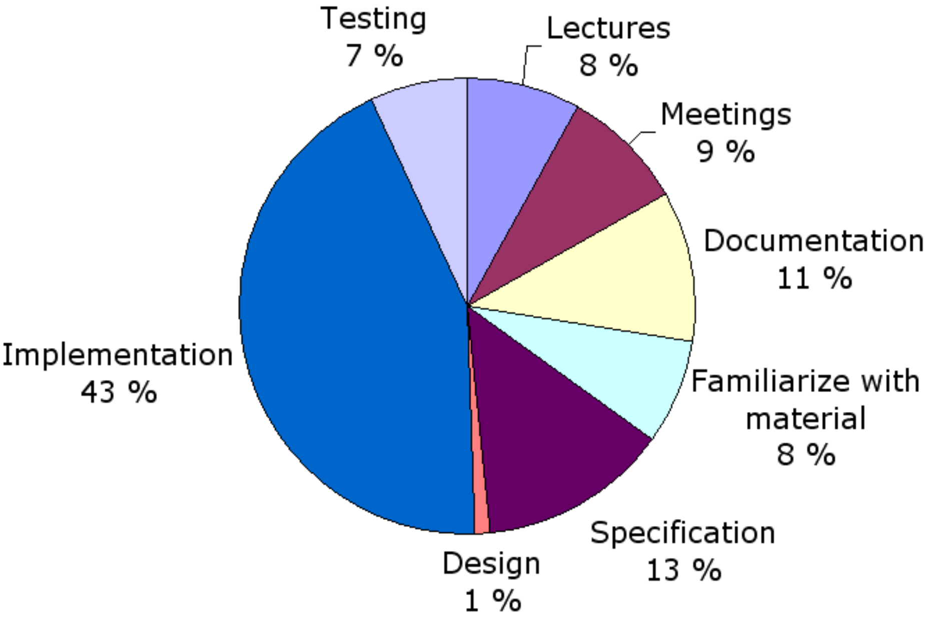


Figure 3: Timo's workload in tasktypes.

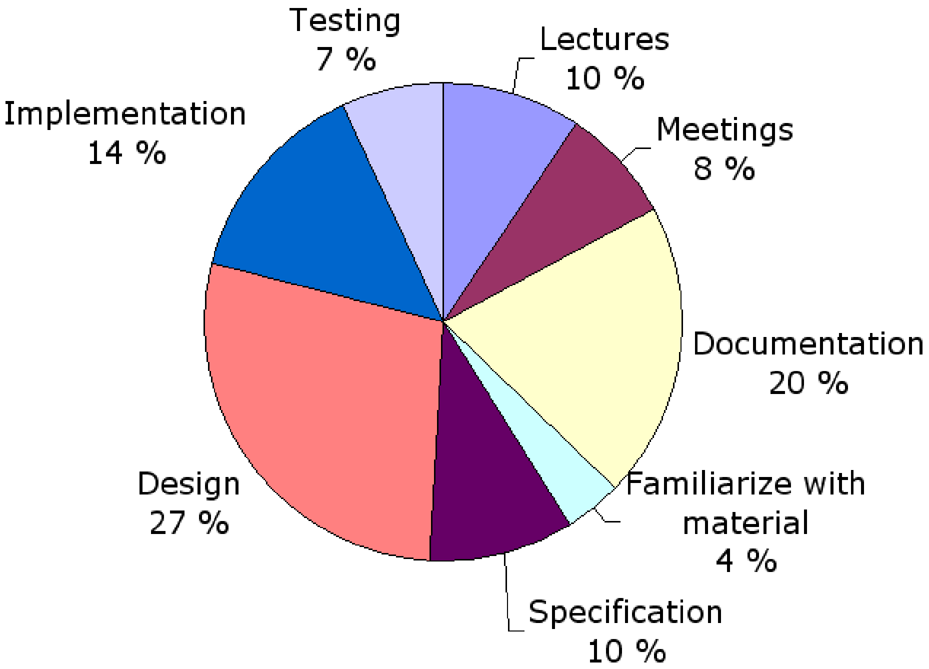


Figure 4: Petri's workload in tasktypes.

Petri concentrated more on the specification, design and documentation, so he spent less time on the implementation. Otherwise the tasks were quite well distributed between the members.

6.6 Workloads on Weeks

The division of the workloads to weeks for each project member individually and all together can be seen in Figures 5 - 9. The figures show how the start was pretty slow but the workloads grew through the project. All the figures present the time usages from week 38 to week 2. The project group also worked sometime after that but those minor workloads are not included to this report.

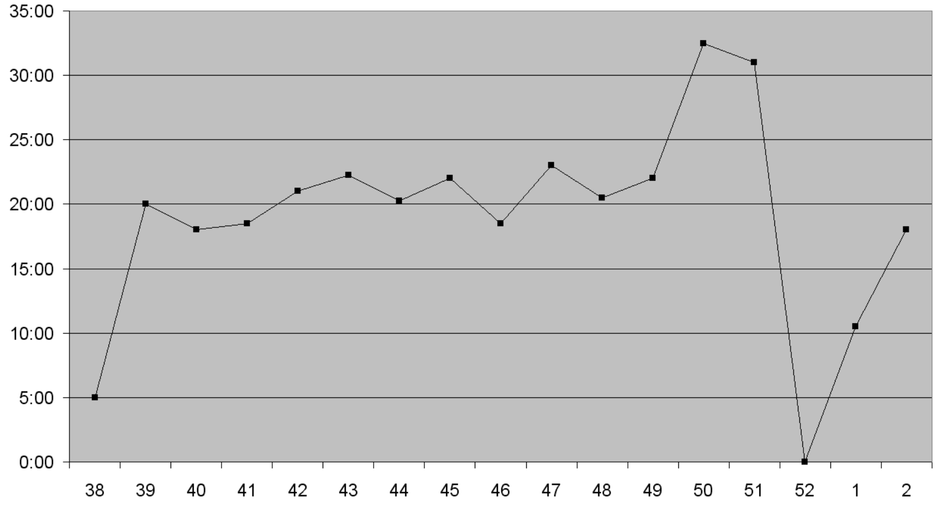


Figure 5: Olli's workload divided to weeks.

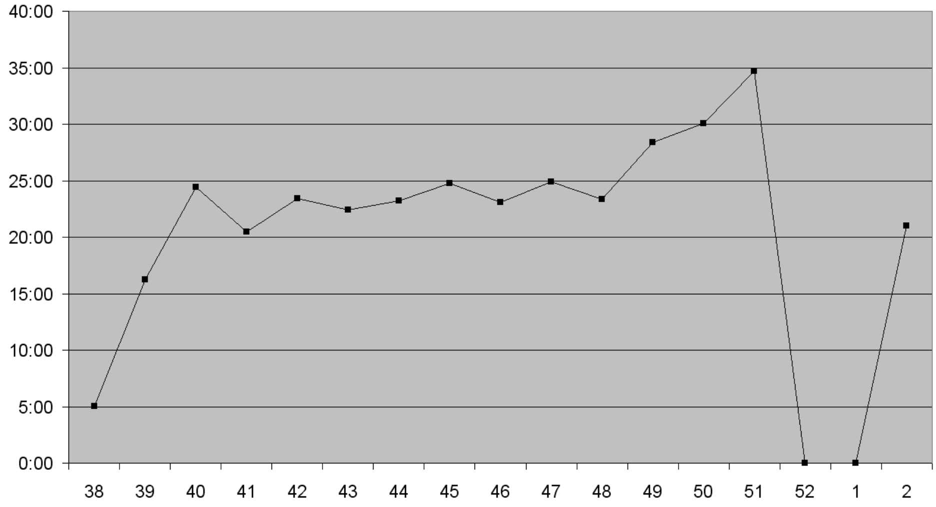


Figure 6: Kimmo's workload divided to weeks.

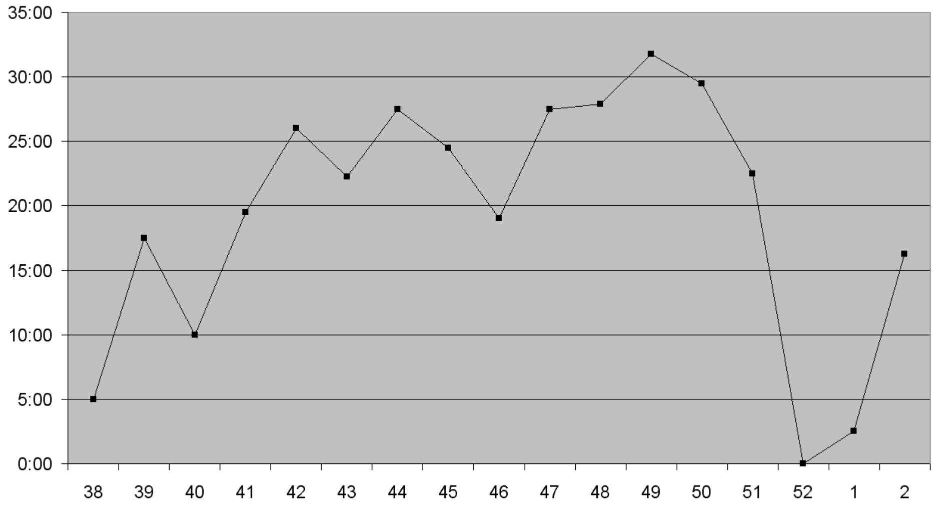


Figure 7: Timo's workload divided to weeks.

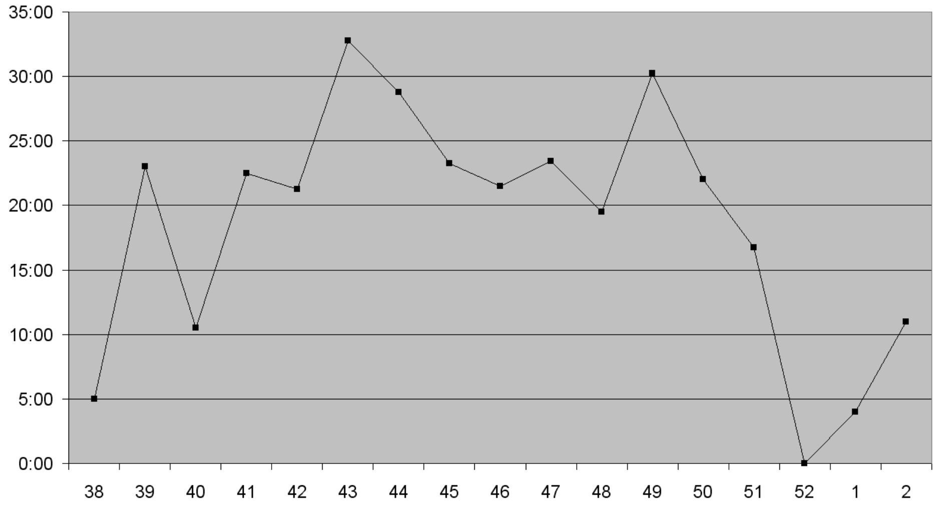


Figure 8: Petri's workload divided to weeks.

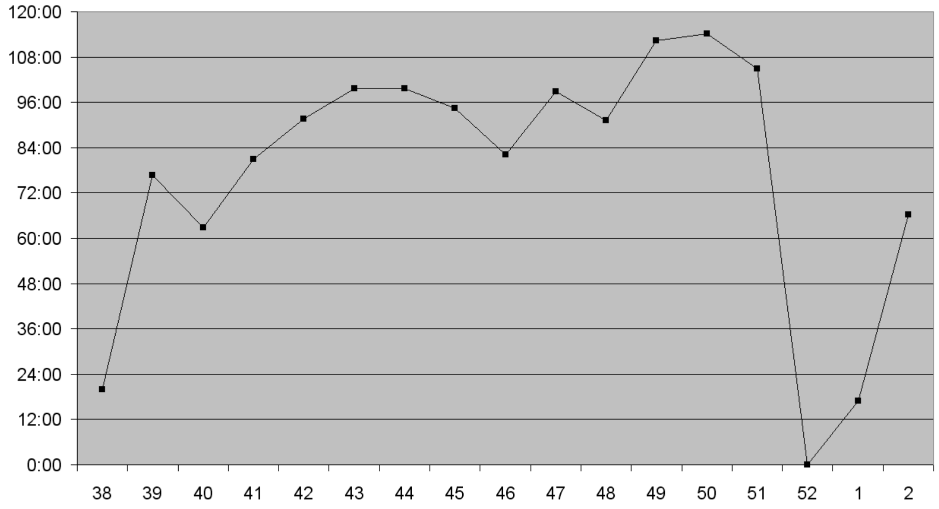


Figure 9: The whole workload divided to weeks.

7 How the Goals Were Reached

The chapter describes how the goals set in the Project Plan were realized.

7.1 Software Goals

The software goals reached can be seen in Software Report. The implemented BlueCheese as a whole was good and matched well to the requirements. The software does work in almost all of the cases and it lacks none of the major features.

The most of the goals specified in the requirements and the specification were reached but some features and bug corrections were left for further development. The stream mode data transferring was the only feature that was totally left out of the implementation. Other unreachable features were at least somehow implemented.

7.2 Educational Goals

The educational goals of the project were reached as the project members learned a lot about software projects and their management in general. The teamwork and the social skills of each member developed and the meeting practices became familiar.

The project members also learned a lot of practical skills. The English vocabulary was expanded and the documentation process in general and with LaTeX became very familiar. In the implementation phase Symbian OS's C++ syntax was learned through the hard way because none of the project members had any experience on it before the project.

8 Realized Schedule

The chapter describes the realized schedule of the project. Comparison between the planned and the realized schedule is also done.

8.1 The Figures

The planned schedule can be found from the project plan. The time intervals of both the planned and the realized tasks can be found from Figures 10 and 11. The project schedule shows all the bigger tasks and the implementation schedule shows the more detailed implementation tasks. The black bar in the figures shows the realized time usage and the lighter bar shows the planned one.

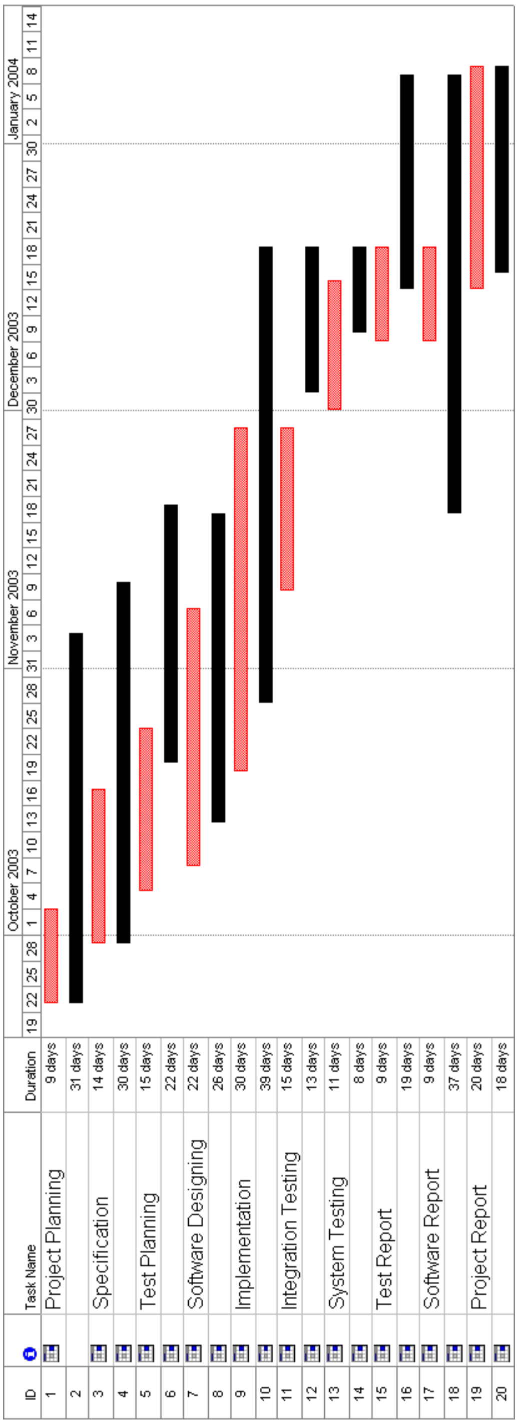
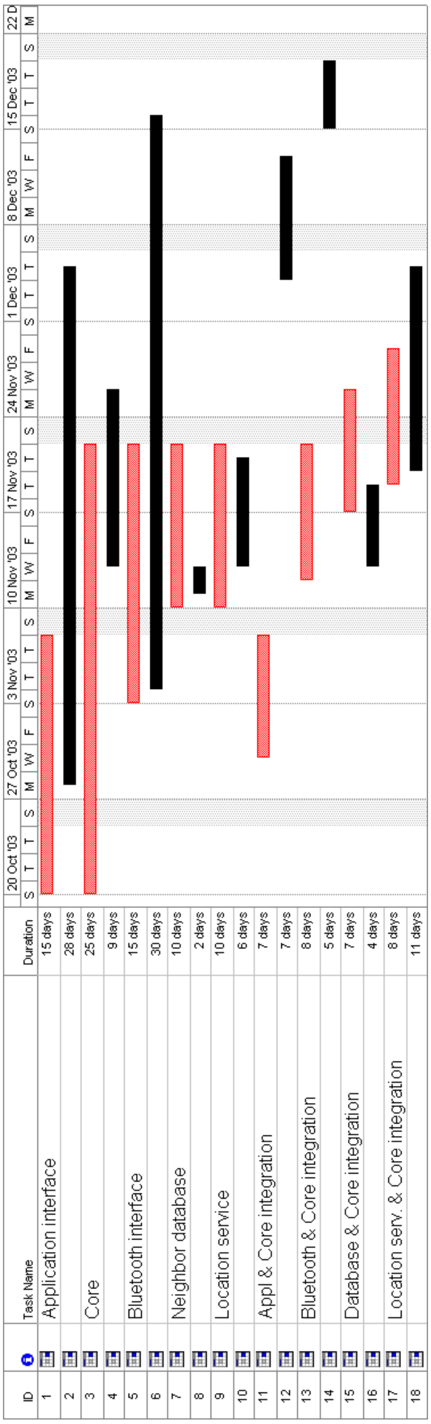


Figure 10: The realized project scheduling by the tasks.



ID	Task Name	Duration
1	Application interface	15 days
2		28 days
3	Core	25 days
4		9 days
5	Bluetooth interface	15 days
6		30 days
7	Neighbor database	10 days
8		2 days
9	Location service	10 days
10		6 days
11	Appl & Core integration	7 days
12		7 days
13	Bluetooth & Core integration	8 days
14		5 days
15	Database & Core integration	7 days
16		4 days
17	Location serv. & Core integration	8 days
18		11 days

20 Oct '03
27 Oct '03
3 Nov '03
10 Nov '03
17 Nov '03
24 Nov '03
1 Dec '03
8 Dec '03
15 Dec '03
22 D
S T T S M W F S T T S M W F S T T S M W F S T T S M W F S T T S M W F S T T S M W F S T T S M W F S T T S M

Figure 11: The realized implmentation scheduling by the tasks.

8.2 The Analysis

At the start of the project, none of the tasks were completed by their deadlines. That was mostly due to the fact that the requirements were not as detailed as we expected. That stretched especially the specification and the design phases and also the starting of the implementation.

It also took some time to understand the concept of what we were doing and how it should be done. English language and the familiarization to the technics also took some extra time during the first weeks.

The designing of the software would have taken the whole project if we only had allowed it. In the middle of the November the designing part had to be ended. The design document was freezed and the further changes of designing were done to the software report. Some changes were done after the freezing and they can be seen from Software Report. The reason for designing taking so much time, was that the implementation set a lot of restrictions to some features. Those restrictions couldn't have been known before trying to implement the features.

At the end of the project the software report's name was again changed to software design. That is why the project produced two different design documents. The version 1.1 is the frozen one and version 2.2 is the final design that used to be the software report.

The beginning of the implementation was also pretty tough. None of us knew the Symbian OS before and a lot of time was again used for reading material. But when the implementation got fully started, we started to reach the schedule. All the project members did a lot of work to gain that goal.

The planning of the implementation was done based on our best knowledge, but that still wasn't perfect. The Application Interface and the Bluetooth Interface parts needed much more time than we expected. The implementation deadline had to be stretched during the project and the testing had to be respectively shortened.

At the end of the project, all the members of the group made a hard work and somehow all the tasks were completed. The system testing was the only task that wasn't done. It was a result of that the GPCS application was not ready early enough.

9 The Realization of the Risks

The chapter describes the realization of the risks and also how they were handled or avoided. The risks expected can be found from the project plan.

The inexperience in the software projects was handled with careful planning and good interaction with the customers and the supervisors. As a result that risk didn't cause too much harm to the whole project. The motivation of the project members was high and the relationships between the members stayed good all the time, which also helped us to carry out the project.

The language used in the project was English and that also took some extra time. The documentation and the meeting reports weren't very quick to write out. Bigger misunderstandings were still avoided and the language didn't became a huge issue.

Practical risks caused probably more harm. Symbian OS's C++ programming was not as easy as one could have supposed. A lot of delays came because of some weird features of it. The C++ was familiar to all the project members but it didn't seem to be enough. The tools of the project also stretched the timetable especially due to the lack of a Bluetooth adapter that Symbian SDK would have accepted. We couldn't debug the communication parts of the software because of that deficiency.

The delayed completion of the test application was also a problem. The GPCS application was not ready by the end of the project as the BlueCheese completed and the API testing was delayed. Anyway, that was a risk that we couldn't control.

10 Project Experiences

The chapter describes the common and the individual experiences of the project members.

10.1 Common Experiences

Generally the project members feel that the project went well. There were some problems of course, but they didn't ruin the project. The customers active participation also helped members to succeed.

Time was probably the biggest problem in the project. Four months was a very short time and Christmas also made it even shorter. All the project members worked hard and we were in great hurry at the end of the project.

Still, all the members are quite satisfied on the experience and the goals that were reached. There would also not be many things that would be done differently if the project would start over now. Of course the knowledge got in this project would help on start, but with the premises we got, not many things could have been done better.

10.2 Olli Alanen

The project's subject was very interesting and it helped to stay motivated from the very beginning. When the project proceeded the interesting subject sometimes seemed to be even too challenging. After all the motivation kept high through the whole project.

The great team spirit helped us to work too. The project members relationships stayed good for all the time and because of that, it was fun to work for the project most of the time.

The project gave a lot to me. I learned much about the general project working methods and project management. The Symbian OS coding was also new for me and it was very interesting to learn it. LaTeX was familiar from the past, but I learned some good skills of the general documentation process.

So, the project was a great experience and I learned a lot. It took a lot of time but I believe it was worth it.

10.3 Kimmo Haukimäki

The project was very challenging in many ways. The project was done in English and that increased workload. But the hardest part in the project was programming part. Symbian C++ syntax is little bit different compared to normal C++ and that is why we needed to familiarize with that.

However the bigger problem in the programming was our subject area. The software was related to telecommunication and none of us has earlier programmed telecommunication software, which was very challenging and it took a lot of time.

Personally I'm very satisfied to the project. I learned a lot of new skills, which I will need in the future. Symbian C++ programming language is very useful skill for software engineering student in ”mobile phone land”. Secondly I'm very happy when I learned communication skills in English. The motivation to the project before the beginning was to get to know what a big project is like. I'm very satisfied to have got that experience during the project.

10.4 Timo Juonoja

The subject of the project was very demanding and challenging, but very interesting. The project completed successfully even though all the requirements weren't implemented because of the very short time interval. However I'm satisfied with the product of the project.

The project members were very hardworking and keen on working as a group. It was a pleasure to work with them. All the members did their own tasks very well.

The software project taught me about project working. The skills of English sharpened and I got familiar with the meeting behaviours. I also learned about programming and hands-on experience of the telecommunication.

10.5 Petri Rönkkö

I think the topic of our software project was very interesting and challenging. Project group worked very hard and were able to finish project in the time what was reserved for the project. The project was handled in English which was very good, because it made the project more international.

The software project taught me to work as a member of the project group. My English improved much during the project and I learned to use new working tools. I think the project was successful and it gave me lots of experience about the software project.

11 Conclusion

The project was a success. The goals were reached and the BlueCheese was implemented with almost all of the features that were required. The planned schedule didn't match exactly to the realized one, but all the tasks were done.

All the project members were very satisfied to the project. The subject was challenging, but the project group managed to carry it out well.

Bibliography

[1] Open Source, ”Academic Free License v. 2.0”, available in PHP form <URL: http://www.opensource.org/licenses/afl-2.0.php>, referenced 19.01.2004.

[2] Avenue 12, “Wireless Bluetooth Specification, Device & Connection Terms & Glossary”, available in HTML form <URL: http://www.website101.com/bluetoothwidget/bluetooth_info.html>, referenced 04.11.2003.

[3] Coulouris George, “Distributed Systems: Concepts and Design, 3rd edition”, Addision-Wesley, 2001.

[4] Kotilainen Niko, Pentti Ville, Rastas Jussi and Töyrylä Joni, ”Guardian Project, Project Plan”, University of Jyväskylä, Department of Mathematical Information Technology, 16.12.2003.

[5] Hole Kjell J., “Bluetooth, Part 7: RFCOMM”, available in PDF form <URL: http://www.kjhole.com/Standards/BT/BT-PDF/Bluetooth7.pdf>, 2.5.2003.

[6] Jipping Michael J., ”Symbian OS Communications Programming”, John Wiley & Sons, Ltd, 2002.

[7] Kam, Matthew, “Hilarius, A P2P Price Gossiping System ”, University of California at Berkeley, 2003.

[8] Kam, Matthew, “Simulation Outline ”, University of California at Berkeley, 2003.

[9] Nokia, “Bluetooth”, available in HTML form at <URL: http://www.nokia.com/nokia/0,8764,397,00.html>, 2003.

[10] Palowireless Bluetooth Resource Center, “RFCOMM Layer Tutorial”, available in ASP form <URL:

http://www.palowireless.com/infotooth/tutorial/rfcomm.asp>, referenced 27.10.2003.

[11] Rastas Jussi and Törmälä Ville, "Get a Lift/Give a Lift Service", University of Jyväskylä, 2003.