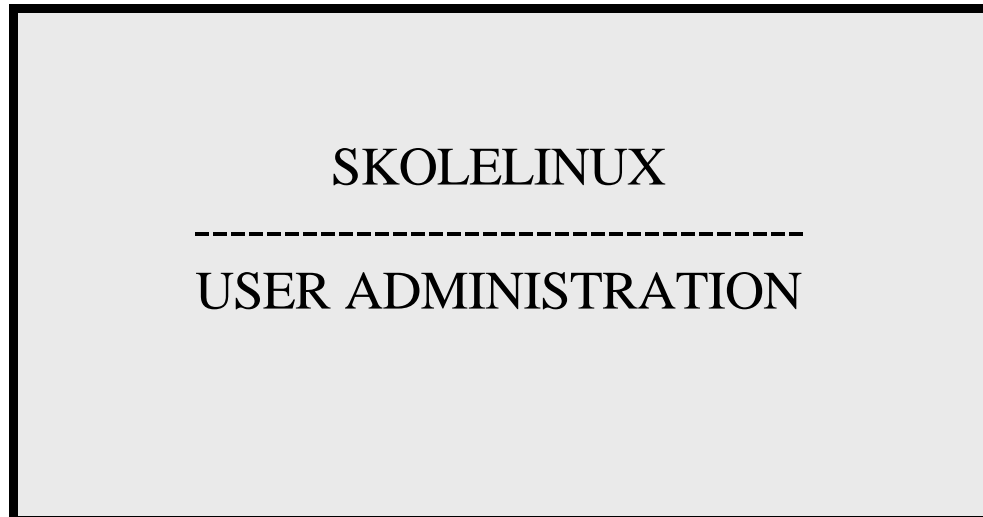


MAIN PROJECT:



AUTHORS:

Trond Christian Frøhaug  
Morten Sporild  
Ole Martin Dahl

Date:

May 19 2003

**SUMMARY OF THE MAIN PROJECT**

Title:	SKOLELINUX – USER ADMINISTRATION	Nr. :	8
		Date :	19.05.03
Participants:	Trond Christian Frøhaug		
	Morten Sporild		
	Ole Martin Dahl		
Teacher supervisor:	Erik Hjelmås		
Employer:	Frode Hjemtland representing Skolelinux		
Contact person:	Frode Hjemtland		
Headwords	GNU/Linux, Printer quotas, Netgroups and LDAP.		
(4)			
Pages: 92	Appendices: 10	Accessibility(open/confidential): open	
Short description of the main project:			
<p>This project deals with improving the user administration features in Skolelinux. The project is a part of the larger Skolelinux project, aimed at making a GNU/Linux distribution for Norwegian schools.</p> <p>The user administration feature this project deals with, is printing quotas (using CUPS and PyKota) and netgroups in a directory service (LDAP).</p>			

## PREFACE

This “main project” is a final project that every student at Gjøvik University College has to carry out in the last semester of the Bachelor of Computer Engineering education. It has a stipulated workload of 18 ECTS points. Our project period has been from January 2003 with a deadline for delivery at May 19. 2003.

The project deals with system administration in Skolelinux. More specific, printer quotas and LDAP netgroups. Skolelinux is an open source software project aimed for Norwegian Schools. Skolelinux is a GNU/Linux distribution providing Norwegian user applications in the two Norwegian dialects, Nynorsk and Bokmål, and in the Sami language.

This report documents the complete work process during the project period.

Gjøvik May 16 2003

---

Trond Christian Frøhaug

---

Morten Sporild

---

Ole Martin Dahl

## CONTRIBUTIONS

During the project period several people have contributed in various ways. These are the people we feel to give attention:

Erik Hjelmås ( our teaching supervisor )

Petter Reinholdsen ( developer at Skolelinux )

Jarle Osmund Vågen ( developer at Skolelinux )

Dagfinn Ilmari Mansåker ( developer at Skolelinux )

Frode Ydse Jemtland ( employer )

## Table of Contents

SUMMARY OF THE MAIN PROJECT.....	2
PREFACE.....	3
CONTRIBUTIONS.....	4
1 INTRODUCTION.....	7
1.1 Background.....	7
1.2 The project group.....	9
1.3 The employer.....	10
1.4 Problem Area.....	11
1.5 Time and project limit.....	12
1.6 About the report.....	13
1.7 How we approached the project.....	19
1.8 GNU/Linux.....	21
1.9 Planning and reporting.....	23
1.10 Development model.....	25
1.11 Ranking of functional demands.....	29
1.12 Requirement specification.....	31
2 MAIN SECTION.....	41
2.1 Literature review.....	41
2.2 The different technologies.....	43
2.3 Increment 1 - Printing quotas.....	54
2.4 Increment 2 - Netgroups in LDAP.....	69
3 CLOSURE.....	77
3.1 Discussion of the project.....	77
3.2 Criticism of the project.....	80
3.3 What we have learned.....	82
3.4 Further work.....	83
3.5 Evaluation of the project group.....	85
3.6 Conclusion.....	88
4 References:.....	89
5 Appendices.....	92
A - Gantt charts.....	
B - Meeting reports.....	

C - Status meeting reports.....

D - Log of hours used .....

E - Technologies .....

F - The Gatherings.....

G - Pykota installation guide.....

H - Pre-engineering project .....

I - Project contract.....

J - CD-ROM.....

# 1 INTRODUCTION

## *1.1 Background*

When a user uses a computer in multi-user environment many services are needed. E.g. the user need individual home areas and profiles, the user need access to different services and so on. User administration capabilities therefore are important for the administrators. It is important to have control over individual user accounts and different services used by the user. Skolelinux is adapted to school settings where budgets are small and system administration expertise is scarce. It is therefore important for Skolelinux to have a complete system with all capabilities ready to use for the local administrator.

In December 2002 our teacher in System Administration, Erik Hjelmås, got a mail from Skolelinux listing different “main projects” that Skolelinux needed to solve. He presented these potential projects to us. Our group soon after decided to go for the user administration tasks. This because we where all eager to learn more about user administration and GNU/Linux. None of us in the project group had much experience with GNU/Linux and system administration, except from our system administration course of 12 ECTS points. This project gave us the possibly to learn much more.

In January 2003 we had a meeting with Frode Jemtland (our employer representing Skolelinux) and Knut Yrvin (project manager for Skolelinux) at a primary school in Nittedal. This was our first introduction to Skolelinux. The primary school in Nittedal used Skolelinux as main operating system at their school. Knut Yrvin presented how the system worked and the

challenges a school environment gives to computer systems.

Knut Yrvin presented how the work methods where in the project and what they needed of functionality from our group. Skolelinux had to change our original task a little bit, this because another group from NITH (Norges Informasjonsteknologiske Høgskole) also had chosen the user administration task. It was decided that NITH would work on improving the graphical user interface (GUI) and that we would work on underlying structure modification of the user administration system. This meeting was the base of our pre-engineering project, delivered January 24 2003. See appendix H.

Skolelinux uses a central directory service server to store user data. E.g. user names, passwords, email addresses etc. This solution is chosen because Skolelinux wants to have a central server where all the user data is stored and make it easy to use. One of our main challenges is to make systems that use this directory service to store all the new user information we need.



## 1.2 The project group

Our project group consists of three students from Bachelor of Computer Engineering at Gjøvik University College, Norway (branch of study, system administration). All in the last year of the three-year program.



Ole Martin Dahl

Ole Martin was born in 1980 and comes from Jessheim.

He has studied at Gjøvik University College, since finishing high school and one year basic training in the army.



Trond Christian Frøhaug

Trond Christian was born in 1979 and comes from Hønefoss.

Before attending Gjøvik University College, he spent one year at Oslo University College and one year with music studies.



Morten Sporild

Morten was born in 1980 and comes from Hamar.

He has studied at Gjøvik University College, since finishing high school and one year basic training in the army.

### **1.3 The employer**

Our employer is Frode Jemtland. He represents Skolelinux [1].

Skolelinux is a project started by the idealistic member organization “Linux i Skolen” [2]. This organization was founded July 16. 2001. “Linux i Skolen” aim to arrange for and inform about the use of open software in Norwegian Schools. It collects all information about open software for schools and distributes this information by use of web, mailing lists and other channels. The organization acts as a forum for discussion and exchange of practical advice and experience between system administrators at schools. It also supports new open software adapted to Norwegian schools with translation and development. With schools they mean primary schools, junior high schools, high schools, colleges, universities and similar educational institutions.

The Skolelinux project is a voluntary project from “Linux i Skolen” and is lead by Knut Yrvin. The project goal is to develop a GNU/Linux distribution that is easy to install, maintain and is translated into the two main Norwegian dialects Bokmål, Nynorsk and the Sami language.

### **1.4 Problem Area**

We were given multiple problems from Skolelinux, which we needed to range according to what we thought were critical and possible to solve. And then choose the tasks that we had the time and ability to solve. These assignments were:

- To implement a system for printing accounting/quotas in the Skolelinux distribution. It must be possible to set an individual quota for each user in the LDAP-database.
  
- Realize netgroups of users and machines in Skolelinux. This has to be done by using LDAP, so that specific machines can grant access to only certain groups e.g. only teachers in a classroom.
  
- Make a solution on how to add multiple users into the LDAP-database. These users will come from a plain text input file, in different formats from the school's existing systems (like Cerebrum).
  
- Automatically create individual e-mail accounts for users as they get created in webmin for the LDAP-database.

### **1.5 Time and project limit**

#### **Time limit**

The project period started when we delivered the pre-engineering project January 24, 2003 to the deadline of report delivery Mai 19, 2003. All the time limits has been written down and continually updated in our Gantt chart. See appendix A.

The project group decided on a workload at a minimum of 24 hours each week. This has been kept and exceeded quite a bit. The workload of this project is stipulated to 18 ECTS points i.e. about 360 hours total or three months of work for each student. One year of study is stipulated to 60 ECTS points. A diary of hours used and what we have done each day is documented. There is one for group work and one individual diary for each group member with individual hours. See appendix D.

#### **Project limit**

The project has been reduced from the first planned description in the pre-engineering report. This because all the different tasks given from Skolelinux proved to be too large for us. The problem area was very open and the problem description was very up to us to decide. The requirement specification was developed along the first months of the project period.

The project is limited to printing quotas in Skolelinux and LDAP netgroups in Skolelinux.

## **1.6 About the report**

### **Target group**

The target of this report is mainly developers and administrators of Skolelinux. It explains technologies and solutions on the use of printer quotas and netgroups in GNU/Linux. The reports technical explanations and solutions may also be interesting to administrators on other GNU/Linux solutions. Future student projects may also need information on the subjects this report cover. A development of a user-friendly graphical user interface upon our printer quota solution and netgroup testing will probably be developed by Skolelinux or maybe some of us in the future, and will use this project to solve their tasks.

The report is documentation for our school and employer on what we have done. Our teaching supervisor and an external sensor use this report as basis for the grade we achieve on this main project.

### **Terminology**

We have chosen to write this report in English. This because it was a wish from our employer to have all development documentation in an international version so other countries could use the work being done here in Norway. We warn about miss spellings and wrong sentences. We have tried to make the report as easy to read as possible.

### **Layout**

The project is written in OpenOffice.org [3]. This is a free software solution. It is a complete office suite that supports many platforms e.g. Linux and Microsoft Windows.

Most text is written in Palatino Linotype fonts. Courier

typewriter fonts are used on hyper links, code and system commands.

The report is divided into five different parts:

**1 Introduction.** This part is about preparation, the report it self, the project work and requirements.

**2 Main section.** This part covers our work, solutions and important technologies regarding our work.

**3 Closure.** This part is about what we have learned, future work and finally concludes the project.

**4 References.** Reference list.

**5 appendices.** This part includes all appendices to this project. These appendices are referred to during the report.

See also the index.

The report also include a CD-ROM with the report and appendixes in .pdf format. It also include our homepage used in the project period. See appendix J.

**Word definitions**

This chapter describes different words and terminologies used throughout the report.

**Directory service**

Not the same as a database pr definition, but work in a similar way.

**Daemon**

A background process.

**Host**

A node in a network that has an IP-address.

**HOWTO**

A document that step by step describe how a technology work and how to set it up.

**Multi-user environment**

Is a computer network with multiple users.

**Netgroup**

A netgroup is a group of users or machines. It is a way to define groups of hosts or users. It can be used to limit access to different hosts for different users.

**Open Source**

Free and available software, distributed as source code.

**User authentication**

Prove that the user is the person he/she claims to be.

**User account**

Consists mainly of a user name and a password.

Often it also consists of other user information e.g. full name, email address, home area, disk quota, printer quota etc.



## Abbreviations

**cn** = common name

**CUPS** = Common Unix Printing System

**CVS** = Concurrent Versions System

**dc** = domain component

**DHCP** = Dynamic Host Configuration Protocol

**dn** = distinguished name

**GID** = Group ID

**GNU** = GNU is Not Unix

**GUI** = Graphical User Interface

**IP** = Internet Protocol

**IPP** = Internet Printing Protocol

**NIS** = Network Information Service

**NSS** = Name Service Switch

**PAM** = Pluggable Authentication Modules

**HTML** = HyperText Markup Language

**LDAP** = Lightweight Directory Service

**LDIF** = LDAP Data Interchange Format

**ou** = Organizational Unit

**PXE** = Preboot eXecution Environment

**RFC** = Request For Comments

**SLAPD** = Stand-alone LDAP daemon

**SLURPD** = Stand-alone LDAP Update Replication Daemon

**SNMP** = Simple Network Management Protocol

**ssh** = Secure Shell

**SQL** = Structured Query Language

**UID** = User ID

**TCP/IP** =Transmission Control Protocol

### ***1.7 How we approached the project***

None of us had really heard about Skolelinux before, so there was a lot of reading to be done before even starting. We had some experience with Debian Linux, which Skolelinux is based upon, after following the course System Administration fall 2002, and some of the structure of Skolelinux was explained at the meeting we had with the project manager Knut Yrvin and Frode Jemtland at Nittedal.

After understanding the architecture of Skolelinux, we had to build the test environment. The room that was lent us by the school, was the old system administration course laboratory. We put up interior walls since we had to share this room with the other Skolelinux group. The small test network that we put up consisted of (see fig 1.1).

- 3 workstations (private computers)
- 1 server (from it-tjenesten (IT administration at our school))
- 1 thin client server (already located in the room)
- 1 thin client (from it-tjenesten)
- 1 firewall (made from a red hat distribution)

After setting up the room we also tested different versions of Skolelinux on the servers and workstations, as they wasn't always stable.

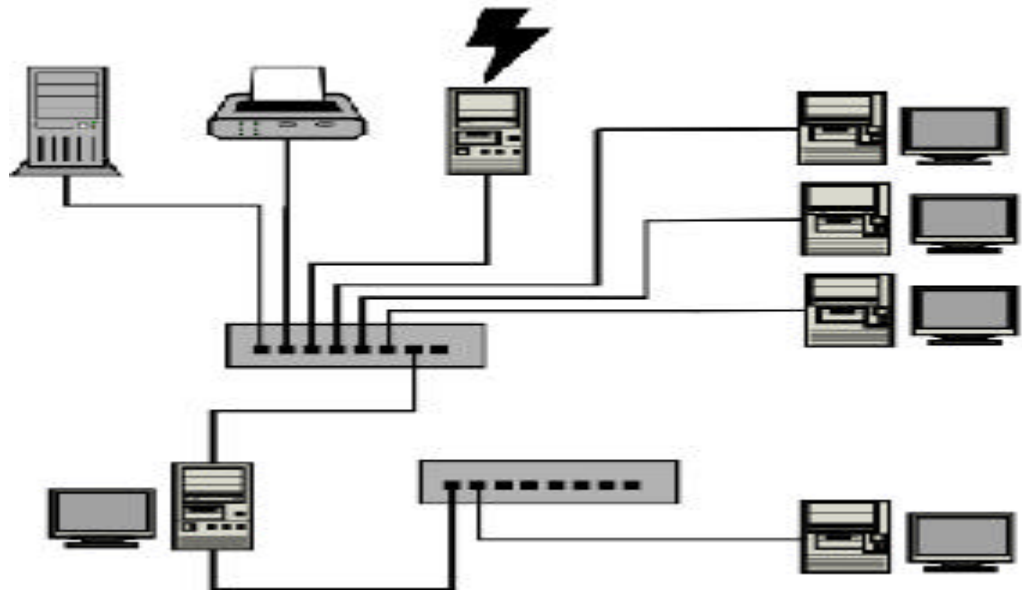


Fig. 1.1 - Our test network

After the meeting in Nittedal, we read the 201 pages pre-engineering report of Skolelinux which was written to “Utdannings- og forskningsdepartementet” in the start of the Skolelinux project. We also read the openLDAP administrator's guide and some documentation on Linux-PAM and NSS. The Internet was also “googled” for existing solutions.

When the test network was up and running, we needed to subscribe the Skolelinux mailing list (`linuxiskolen@skolelinux.no`). This is where all information is given and questions are answered quite soon. The most active developers answer our questions usually during the day. The mail frequency at this list differs from 20-50 each day.

## 1.8 GNU/Linux

Linux is an operating system. Linus Torvalds invented it when he studied at the University in Helsinki, Finland. Linus where interested in Minix, a small open UNIX operating system developed for educational purposes. Linus decided to make a similar but more complete operating system. In August 1991 he released version 0.01 of Linux. The operating system was free for all to use and could be downloaded from a FTP server. Linus also sent emails to news groups about the release of Linux and wanted feedback. Already after few hours the first interests replied. And soon hundreds of programmers conducted on the development of Linux.

At the same time a movement called the Free Software Foundation was finishing a project called The GNU Project. This project was going to secure the freedom to use and change software if people wanted to do so. The goal of GNU became to make a complete operating system like UNIX, but entirely based upon open source. GNU had almost every component ready for the new operating system, but failed to make one important piece: the kernel. Therefore it got agreed upon that the kernel from Linux would be combined with the GNU system and together it would be a complete operating system. This complete system was called GNU/Linux. In 1994 version 1.0 of GNU/Linux was ready. To day mostly called just Linux.

Today Linux is often used as servers on the Internet and in networks. It has been known to be very stable. Gradually it has also become more and more popular to use Linux as a desktop operating system, thanks to new and easy to use graphical user interfaces, like Gnome and KDE. Today Linux has become an excellent operating system with much lower costs than

competing systems, e.i. Microsoft Windows.

## ***1.9 Planning and reporting***

### **Main classification of the project**

The project was divided into four sections:

#### **I Pre-engineering**

Establish contact with the employer.

Preparing the group room.

Build "Skolelinux" test network and install the latest release.

Pre-engineering report.

Conferences with our teaching supervisor.

Gather information.

Make website.

#### **II Research**

Find information and learn about technologies.

Formulate requirement specification document.

#### **III Main section**

Developing.

Testing in our own test network.

Report bugs.

Troubleshooting.

Feedback from teacher supervisor and employer.

Continuously update the project's website.

#### **IV Project closure**

Finalizing the project report.

Presenting of the project.

### **Demands of status meetings and points of decisions**

We've had one meeting with our teaching supervisor every second Tuesday to discuss status of the project, problems and other challenges.

Throughout the project period we have decided to have 6 status meetings with our teaching supervisor. From three of these meetings we will write status reports that will follow a given standard from Gjøvik University College.

Status meeting dates:

- February 4, 2003
- February 18, 2003 followed up by a status meeting report.
- March 4, 2003
- March 18, 2003 followed up by a status meeting report.
- April 1, 2003
- April 29, 2003 followed up by a status meeting report.

See appendix B for meeting reports and appendix C for status meeting report.

Major decisions regarding the tasks will be taken in agreement with our employer. Meeting reports will be written in turn by each group participants.



### 1.10 Development model

Our development model bears the stamp of open source development. A definition of open source development [4]:

*The basic idea behind open source is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.*

Open source or free software is about freedom, not product price. The source code is open for everybody. This gives the user total freedom to use, redistribute, study or change the software. Although most open source software comes with a license of some kind that set limitations and demands to use and/or redistribution.

The developers of Debian GNU/Linux were the first to define this in The Debian Free Software Guidelines (DFSG) with Bruce Perens as project leader [5]. DFSG was a “social contract” with a collection of guidelines that the Debian developers had to follow. This contract formed the basis of a more precise definition of what really open source is, called Open Source definition (OSD) [6], defined by Peres and Raymond in The Open Source Initiative.

The Skolelinux project is open source. Skolelinux has Debian GNU/Linux as basis system. The project is all about voluntary work, and producing is very important. The CVS (Concurrent Versions System) is an important part of the Skolelinux project. See appendix E. All work produced are uploaded to a

centralized server. In this way the last version of all the work done is always available for the developers. Every update made in the CVS is followed up by a mail to the mailing list `commits@skolelinux.no`. By reading this mailing list, every developer can see the latest changes done on the software. If faulty code is detected it is possible to get the changes undone. Another important tool in the development process in Skolelinux is the Bugzilla, a [ 7 ] “class of programs called "Defect Tracking Systems", or more commonly, "Bug-Tracking Systems". This is a program that help developers, individually or in groups, to keep track of special bugs in the system and assign bugs to themselves. This means you can get “the ownership” of bugs, so other people in the project know who's working on them. This is a tool, which is very important in the Skolelinux project, and other open source projects.

Skolelinux arrange gatherings for the developers several times a year. These gatherings organize future work, and are a place where developers can discuss their problems and solutions. We have participated in two gatherings during this project period. See appendix F.

The incremental system-engineering model is based on dividing the project into multiple small projects each with individual increments. See fig 1.2. Each increment consists of development, testing, integration and validation.

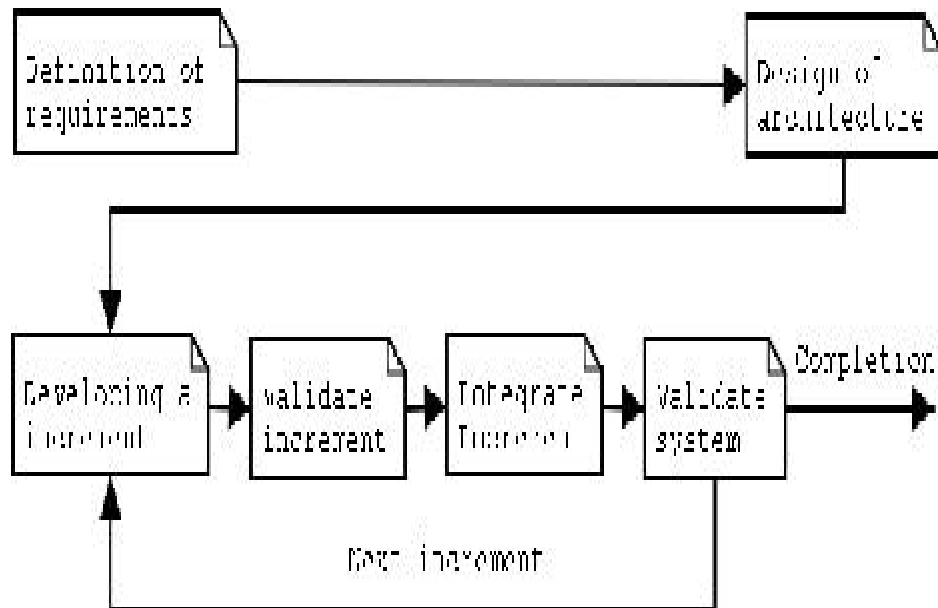


Fig 1.2 - Incremental system engineering model

We started with three increments in our pre-engineering project. After a ranking of our functional demands we decided to reduce our increments to two. The increments were named: printing quotas and netgroups in LDAP. We started with the printing quota increment. In March we also started the second increment in parallel to the first.

We early understood that keeping a structured system-engineering model in this project was difficult. We choose an incremental system-engineering model in our pre-engineering report. As far as possible we have also tried to follow this model. However we discovered that some initially planed goals had to be changed.

The initially planed system specification was early discovered to be difficult to write. We had to research a lot about our project goal ourselves, and therefore the specification got finished during the development. The employers demand was very open, and we had to find out a lot of what was needed to be fixed and how to do it ourself. The incremental system engineering model itself, also support this because each increment has its own

specification. The plan for finishing the system specification at January 24th, was not possible.

One of our initially planed increments was discarded (mail implementation). Because the solution to the increment was started during the first developer seminar by other developers. We realized that the two remaining increments would be enough to work with. Our employer also agreed upon this.

### **1.11 Ranking of functional demands**

During the project we, in cooperation with teaching supervisor and contact person in Skolelinux, found out that we needed to rank our functional requirements. Since we not could realize all the tasks due to time restrictions.

After discussing pros and cons back and forth in the group, we ranked them and decided to go for the two most interesting and most fitting according to what we had read about in January and February. The list came out like this:

#### **1. Printer quotas Pykota and PostgreSQL**

-Pro:

- Needed.
- Started working on this.
- Future versions will support LDAP back end.
- PostgreSQL is needed in Skolelinux for other purposes.

-Con:

- No LDAP back end at this time. Need an implementation of a PostgreSQL database.
- Larger task then first expected.
- Only in 1.0 version.
- Hard to install automatically.

#### **2. Netgroups NSS and LDAP.**

-Pro:

- Started working on this.
- Very interesting.

-Con:

- No working solution has been found to

this problem.

-A lot to learn about.

### **3.Adding multiple users from user lists.**

-Pro:

-Needed.

-Is possible solving with Perl.

-Con:

-Time limit.

-Mostly adapted to the NITH student group that work with the user administration GUI. Was probably their task from the beginning, have been some confusion around who this task was assigned to.

### **4.LDAP/Limacute mail configuration.**

-Pro:

-Solutions do exist.

-Con:

-Is being developed by others at Skolelinux. Andreas Dahl, one of the developers for Skolelinux, mentioned that this was very close to be solved.

-Time limit.

## **1.12 Requirement specification**

### **Introduction**

The requirements of this project were very open for us to specify. The problem area was open to interpreter. From our ranking of the different functional demands given by Skolelinux, we ended up with two main areas to solve. This functional demands are described in 1.11.

This requirement specification forms the starting point for future work on this project. The specification describes system demands that form the basis of solutions and methods during the project period.

This requirement specification will include all the increments we have chosen. The specification will be developed more between the increments.

Our increments are:

1. Printing quotas in LDAP.
2. Netgroups in LDAP

### **Basic demands to the system**

#### **General**

The printing quota system should make it possible for administrators to set printing quotas on users. It is preferred that the quotas are defined in the LDAP directory service. And it must be possible to develop an administration tool in Webmin that builds upon our solution. When a quota is reached the current user should not be able to print anymore unless a new quota is given.

The netgroups must be implemented in LDAP. Groups of machines and users must be defined in a LDAP entry. Then it

must be possible to limit access to machine groups, i.e. a user group has access to a certain machine group, and no other users have access to these machines. An administration tool through Webmin must also be possible to develop.

### **Users of the system**

The user of maintaining the system is mainly system administrators at the different schools and developers at Skolelinux. By implication all the users use the system when they print and is part of a netgroup.

### **Open Source**

All the development/implementation that we do must follow open source. That means that others and we are free to see and change the source code. All the software must follow the GNU General Public License[8].

### **Limitations and assumptions**

The most important is to make it possible to limit the use of printers. Each user should have her/his own printing quota. CUPS support a mutual printing quota for every user but not different quotas for each user. Our goal is to make individual quotas possible.

The printing system that must be used is CUPS.

The most important in netgroup solution is that it must be limited to use LDAP.

The GUI part of the printer and netgroup administration is not our task. A student group from NITH (Norges Informasjons Tekniske Høgskole) is making this interface.



## **System description**

### **Structure in Skolelinux**

As mentioned earlier Skolelinux is based on the Debian distribution of GNU/Linux. This was chosen because Debian intends to be the most stable GNU/Linux distribution. GNU/Linux also comes in many other distributions, for example: Mandrake, Red Hat and SuSE.

Debian is the most conservative distribution of them all. In other words it do not implement any functionality unless it is well tested.

Another reason is the package system that supplies Debian with new, and updated packages. It is easy to use with the “apt-get” commands, and offers at the moment nearly 9000 packages.

The network architecture of Skolelinux is described in fig 1.3. Some important technologies used in the Skolelinux network are described in appendix E.

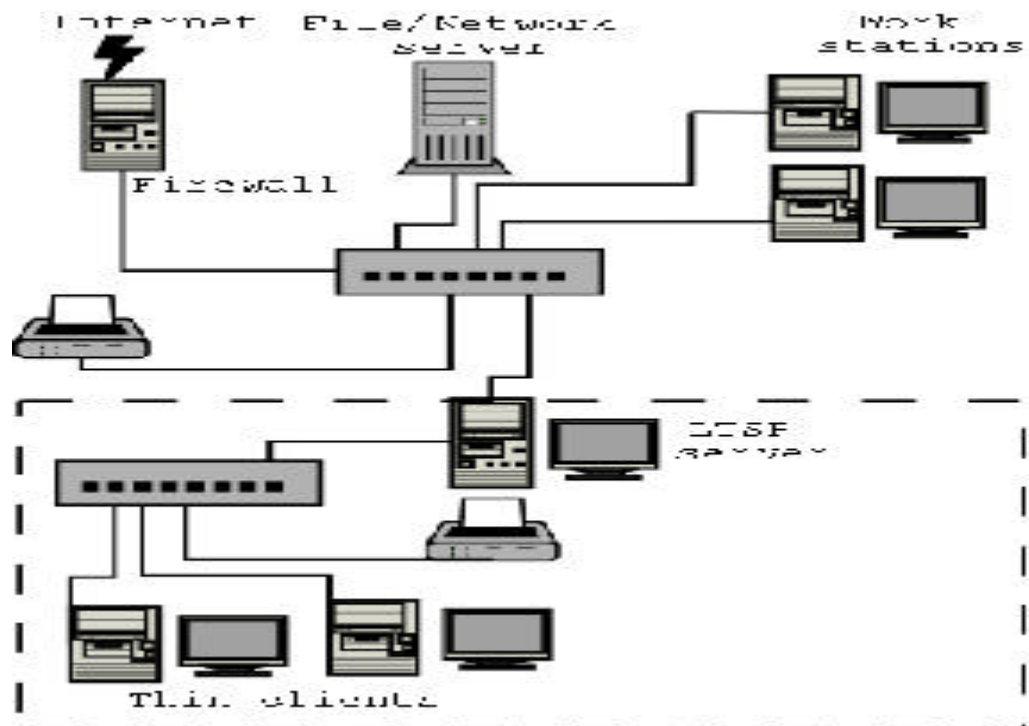


Fig 1.3 - Skolelinux network architecture

To support this architecture the Skolelinux distribution consist of the three following installation profiles:

- Workstation
- LTSP-server ("brain" for the thin clients)
- Main server (file server, mail server, LDAP)

### Workstation

This is for PC's with a "normal" amount of CPU-power, RAM and HD-space. The PC can be connected to the main server, or work as a standalone workstation without connection to the network.

### LTSP-server

This PC must have a great amount of CPU-power and RAM because it will serve as a brain for all the other thin client's.

#### Main server

This PC will serve both workstation and LTSP-servers. Among the many services it offer we will find: file server, mail server, gateway to Internet, ntpd, DHCP server, web server.

The LDAP daemon runs on this server and has entries for every user. The LDAP database contains some information about the user and we want to add information about the printer quotas and netgroups.

All these profiles are on the same CD and under the installation the user choose what he wants to install under the installation procedure.

Our solutions must support the structure of Skolelinux.

#### **Hardware demands**

##### Workstation:

Processor (CPU): 150 Mhz

Memory (RAM): 64 MB

Hard disk: 1 GB

##### Thin client:

Processor (CPU): 80 Mhz

Memory (RAM): 24 MB

##### LTSP server:

Processor (CPU): 300 Mhz

Memory (RAM): 256 MB

Hard disk: 2 GB

CD-ROM.

NICs: 2

HUB/Switch:

Speed: 10 Mb/s

Ports: 4

### **Software**

CUPS version 1.1.14

LDAP version 3

Skolelinux version pr. 37

### **Services**

All services must be controlled from the main file server, except operating of the thin clients, which the LTSP server controls.

The most important services for our project:

- The file server distributes the file system over the network. It offers the home area for all the users through NFS (Network File System) for GNU/Linux clients.
- IP numbers to the client's is distributed by DHCP.
- CUPS is used as printing system.
- PostgreSQL is used as database system.
- OpenLDAP is used as directory service.
- Apache is used as web server.
- Date and time is synchronized by NTP (Network Time Protocol)

-DNS (Domain Name Service) is used as name service.

### **System administration**

It must be possible to administrate all hosts and services installed in Skolelinux from a single host. This also include our solutions. Preferably from the file server or through ssh.

### **Detailed requirement specification**

#### **Structure**

A user's entry in the LDAP-tree looks like this at the moment:

```
dn: uid=tc,ou=People,dc=skole,  
dc=skolelinux, dc=no  
objectClass: posixAccount  
cn: Trond Christian Frohaug  
uid: tc  
uidNumber: 10001  
gidNumber: 10001  
homeDirectory: /skole/tjener/home0/tc  
loginShell: /bin/bash
```

We wish to add another field in this entry that contains information about each users printing account. This can be an integer that is decremented every time the user prints a page.

We then wish to use a new Organizational Unit (ou) like People that include the support for Netgroups. This entry will look like this:

```
dn:  
cn=sysadmins,ou=netgroup,dc=skole,dc=skolel  
inux,dc=no  
cn: sysadmins
```

```
nisnetgrouptriple: (,morten,)  
nisnetgrouptriple: (,user2,)  
nisnetgrouptriple: (,user3,)  
nisnetgrouptriple: (,user4,)  
nisnetgrouptriple: (,user5,)  
objectclass: top  
objectclass: nisNetGroup
```

### **Demands to system support**

All the different machine profiles in Skolelinux must be supported.

The netgroups must include the LTSP servers as well as the workstations.

### **Possible technologies and solutions**

#### **Printer quotas**

##### **CUPS Version 1.1.x**

This version is the current official version. This version does not have LDAP support. It also has a very simple way of dealing with quota limits. A single set of limits applies to all users for the printer concerned. It is not possible to separate users in different groups. All users must have the same amount of prints.

##### **CUPS Version 1.2.x**

This version is under development and is only available through CVS. This version however will support LDAP and maybe the beta version now available does support LDAP. This version will also have a more advanced quota system.

#### **Pykota**

Pykota, is Free Software distributed under the GNU General Public License. Pykota support CUPS, and quotas for each user are supported. So this software may solve the printer quota

problem. However it is not sure that LDAP support is possible to implement. LDAP is not supported in the current version (0.96). If it were possible to make Pykota authenticate users and store quota information in LDAP this would be a good solution. Pykota is only available by CVS in a beta version.

### **Netgroups**

#### **LDAP**

Use the NSS\_LDAP and PAM\_LDAP to make the netgroups work. These modules is developed by PADL Software Pty Ltd [9]

### **Functional specification Administration**

It must be easy to change the print quotas for each user. It must be possible to view each users quotas and how much paper that have been used. The administrator must be able to make individual quota account for each user. It must also be possible to set a new quota limit. A default print policy must also be set: deny or allow.

It must be possible to make a Webmin module to administrate the netgroups from our solutions. This is a requirement because Skolelinux wish to have all administration features through the Webmin interface.

We have to collaborate with the NITH group that is making the GUI.

### **Updating**

All updating of software concerning our implementation should work fine. If problems arise it's important that our solutions is well documented and is possible to change our implementations to support new versions of CUPS, LDAP, Pykota etc.

**Alteration in the requirement specification**

A requirement specification forms the basis of future work in a project. It should not be interpreted as a unchangeable template that must be followed, but as a recipe that can be changed along the project period.

In our project some of the increment requirements has been changed. See the closing chapters for more on this.



## 2 MAIN SECTION

### 2.1 Literature review

When we first started this project, we must inform that our knowledge in the field of GNU/Linux, especially the area of user administration was just basic. Our experience and knowledge with this came from a 12 ECTS point system administration course. In this course we learned the basics of system and user administration. A lot of new exciting areas had to be learned. We started to study and read about the new (for us) technologies early in January. We search the Internet and ordered books/articles via the college library. You will find many of this sources referred to in the reference list. The main sources for our learning curve has been:

- OpenLDAP organization web page.

<http://www.openldap.org>

- Linux online Webpage.

<http://www.linux.org>

- Linux kernel archives.

<http://www.kernel.org>

- Essential System Administration, third edition by Æleen Frich. Published by O'Reilly & Associates, Inc.

- Managing NFS and NIS, second edition by Hal Stern, Mike Eisler & Ricardo Labiaga. Published by O'Reilly & Associates, Inc.

- Mailing lists for Skolelinux, openLDAP and CUPS (news).

- And of course a lot of other pages on the net using Google. <http://www.google.com>

Important questions had to be understood: What is and how do

LDAP, PAM, CUPS and netgroups work? We will try to answer this in the following chapters.

## 2.2 The different technologies

### Directory services

A directory service is a specialized database. It is optimized for reading, browsing and searching. There are many different ways of providing a directory service. Different methods allow different kinds of information to be stored in the directory, different requirements to access information, how it is protected from unauthorized access etc.

There are both local and global types of directory services. Local providing service to a restricted context. Global are usually distributed, meaning that the information (data) is spread over many machines (e.g., the Internet). In a global service the data is often transparent, which gives the same view of the data no matter where you are in relation to the data itself.

DNS (Domain Name System) is an example of a globally distributed directory service. While finger (Linux command) is a local directory service (returns information on users logged on to a single UNIX-machine).

A directory service is similar to a database but has some differences that set them apart from a database[10]:

- Directories are organized in a hierarchical and object oriented way. Information about an object (e.g. a user) is stored in an entry that represents this object in the directory.
- Directory services provide a common schema for what must or can be stored for a certain class of objects.
- Directory services offer a security model. E.g. access restrictions can be specified for one entry and then inherited by all entries below this entry in the tree.

Fig 2.1 shows the hierarchical tree structure of the directory service in Skolelinux

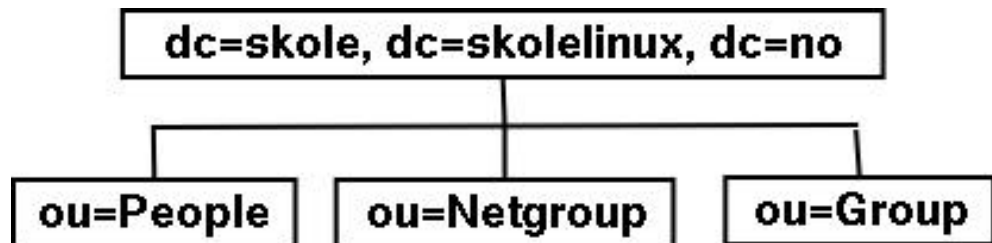


Fig 2.1 - LDAP tree structure

The information stored in the directory service is build up on objects that consist of attributes with values. These objects are defined in schemas.

The most common areas of applications for directories are white pages and yellow pages service. In white-pages services like a phone book, information about an object is accessed by object's name. In yellow pages information is allowed searched/browsed by specifying a category. Because of their flexibility directory services are being used in other applications as well. E.g. as information repository for resources and users, in computer networks.

X.500 [11] is an international standard for directory services. X.500 defines Directory Access Protocol (DAP). DAP is a heavyweight protocol that demand a full implementation of the OSI-model, fig 2.2.

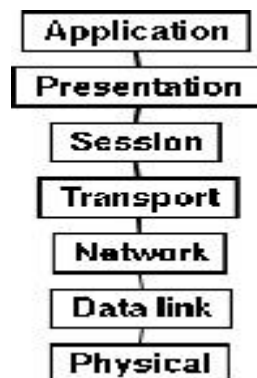


Fig 2.2 - OSI model

The OSI-model has two more layers than TCP/IP, and therefore an implementation of X.500 is very resource and cost demanding. But it exists many implementations of directory services that build upon X.500. E.g. OpenLDAP, LDAP, Active Directory (AD) and eDirectory. The directory services that Skolelinux use, LDAP, run directly over TCP (transport layer) in the OSI model bypassing the session and presentation layers.

### LDAP

LDAP or Lightweight Directory Access Protocol often is called lightweight X.500. LDAP was made because of DAP (X.500) complicity (See directory service chapter). Since LDAP builds upon TCP/IP that is used in most networks today it is more adapted and accessible for the common public.

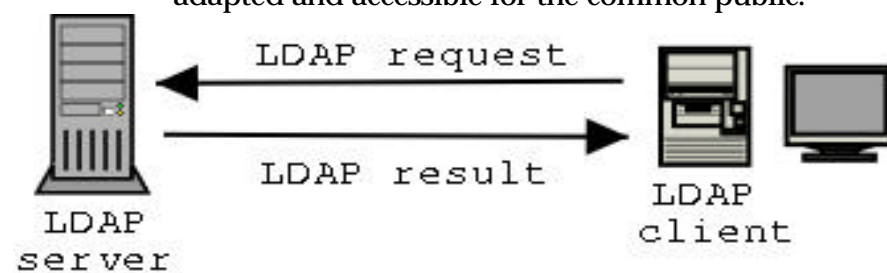


Fig 2.3 - LDAP client/server model

LDAP is based on a client-server model. This makes it possible to access directory information from any computer connected to the network. LDAP work in the way that a client requests

information from the server across the network, the server receives the requests, process it and returns the result. See Fig 2.3. It is also possible to run client-server operation on the same machine.

LDAP can be described as a database used for transaction processing. The special characteristics that is different from a common database is:

- It is optimized for reading.
- It provides advanced searching features.
- It's fundamental data structure, known as the schema, can be extended for local needs
- It's inter portable between different vendors implantations.
- It can take advantage of distributed storage and data-replication techniques.

The contents of the LDAP directory are imported and exported using LDIF (LDAP Interchange Format) files. This is a standard format to read the contents of the directory in ASCII text. This makes it possible to dump the directory content to a readable file. Or write new entries to the directory in a text file and import into the directory.

In Skolelinux LDAP server software in use is OpenLDAP. This is the most popular directory service for GNU/Linux. The OpenLDAP project is dedicated **[12]** "to develop a robust, commercial-grade, fully featured, and open source LDAP suite of applications and development tools". Open basically means that distribution, use and modification of the software is allowed as long as the original copyright holder is properly credited and all warranties regarding the functionality of the software are disclaimed.

The OpenLDAP package includes daemons, configuration files,

startup scripts, libraries and utilities. This is the most important OpenLDAP components:

-**Daemons.** `slapd` is the OpenLDAP daemon, and `slurpd` is the data replication daemon. The replication daemon is used if multiple LDAP servers are in use. The daemon sees to that two or more LDAP servers have the same information stored in the directory at all time. `slurpd` is not in use by Skolelinux.

-**Directory entry-related utilities.** The utilities are `ldapadd` and `ldapmodify` (add and modify directory entries), `ldapdelete` (delete directory entries), `ldapsearch` (search directory entries matching specified criteria), and `ldappasswd` (change entry password).

-**Other utilities.** E.g. `slappasswd` to generate encoded passwords.

-**Configuration files.** Configuration files is stored in `/etc/ldap/`.

OpenLDAP offer a wide range of security features . It can be set up to use libraries from OpenSSL [13] and Cyrus SASL [14] to support transport layer security.

The configuration of OpenLDAP is done in plain text files. The SLAPD server is configured in the file `slapd.conf` (`/etc/ldap/`) on the server and `ldap.conf` on the client.

The LDAP directory are all logically tree structures, see fig 2.1. The root of the tree is most often constructed from the site's domain name like this:

```
dc=skole,dc=skolelinux,dc=no
```

Each component of the domain name becomes the value for a `dc` (domain component) attribute, and all of them are collected into a comma-separated list. This is known as the directory's base. In the case above corresponding to `skole.skolelinux.no`.

An entry in the LDAP database may look like this:

```
dn:
uid=oledahl,ou=People,dc=skole,dc=skolelinux,dc=no
objectClass: posixAccount
objectClass: imapUser
cn: Ole Martin Dahl
uid: oledahl
uidNumber: 10001
gidNumber: 10001
homeDirectory: /skole/tjener/home0/oledahl
description: Elev
mailMessageStore: /var/lib/maildirs/oledahl
userPassword: e2NyeXB0fTViNy41ZVo5dEFDb2c=
loginShell: /bin/bash
creatorsName:
cn=admin,ou=People,dc=skole,dc=skolelinux,dc=no
createTimestamp: 20030505205221Z
```

This data format is known as LDIF (LDAP Data Interchange Format). It's organized as a series of attribute and value pairs. E.g. the attribute `homeDirectory` has the value `/skole/tjener/home0/oledahl`.

The first line is special. It specifies the entry's distinguished name (`dn`), which functions as a unique key inside the database. It's constructed as a comma separated list of attribute values. In the case above, the entry is for user id (`uid`) `oledahl`, organizational unit (`ou`) `People`. The first component of this line is called the entry's relative distinguished name. In the example above this is the `uid=oledahl`. It defines the location within



the sub tree (See fig 2.1) where this entry is stored. The relative distinguished name must be unique within its sub tree. Just as the `dn` is unique within the entire directory.

The `objectClass` attributes specify the type of record: in this case `posixAccount` and `imapUser`. Every entry has to have at least one `objectClass` attribute. Valid record types are defined in the directory's schema, and there are a variety of standard record entries that have been defined.

The directory schema is the collection of object and attribute definitions, which define the structure of the entries. The schema definitions are stored under the `/etc/ldap/schema/` directory. The OpenLDAP package provides all of the most common standard schema, and it's possible to add additional schemas if necessary. The specification on which schema that is in use is specified in `slapd.conf` on the LDAP - server. (`/etc/ldap/slapd.conf`).

### **Linux-PAM**

Linux-PAM (Pluggable Authentication Modules for Linux) is a group of shared libraries that enable the local system administrator to choose how applications authenticate users. **[15]**

This can be done without recompiling the applications, and it makes it possible to switch between different authentication mechanisms. The purpose of PAM is to separate the development of privilege granting software from the development of secure and authentication schemes. This means that PAM enables programs to transparently authenticate users, regardless of how or where user information is stored. The server holding this information could even be located on another continent.

PAM can be used to deny or allow certain programs to authenticate users, certain users to be authenticated or to deny

login from specific users.

Configuration of PAM on UNIX machines is done by editing the `/etc/pam.conf` file and the `/etc/pam.d/` directory with all its files.

These configuration files have the following syntax **[16]**:

```
type control module-path module-arguments
```

The type token tells PAM what type of authentication is to be used for this module. Types of token are:

`account`: Determines whether the user is allowed to access the service, whether their passwords have expired, etc,

`auth`: Determines whether the user is who they claim to be, usually by a password, but perhaps by a more sophisticated means, such as biometrics.

`password`: Provides a mechanism for the user to change their authentication.

`session`: Things that should be done before and/or after the user is authenticated.

The control token tells PAM what should be done if authentication by this module fails. The control types are:

`requisite`: Failure to authenticate via this module results in immediate denial of authentication.

`required`: Failure also results in denial of authentication, although PAM will still call all the

other modules listed for this service before denying authentication.

`sufficient`: If authentication by this module is successful, PAM will grant authentication, even if a previous required module failed.

`optional`: Whether this module succeeds or fails is only significant if it is the only module of its type for this service.

The `module-path` tells PAM which module to use and (optionally) where to find it. Most configurations only contain the module's name, as is the case in our login configuration file. When this is the case, PAM looks for the modules in the default PAM module directory, normally `/usr/lib/security`. However, if your Linux distribution conforms to the Linux file system standard, PAM modules can be found in `/lib/security`.

The `module-arguments` are arguments to be passed to the module. Each module has its own arguments. For example, in our login configuration, the `"nulok"` ("null ok", argument being passed to `pam_unix.so` module, indicating the a blank ("null") password is acceptable ("ok").

The library that gives programs access to Linux-PAM is :  
`/lib/libpam.so.*`.

When using services in Skolelinux, the user has to be authenticated against the LDAP directory. To use PAM and LDAP together a module is required, `pam-ldap`. This module is developed by Padl software [17]. E.g. SSH connections, the system uses the file `/etc/pam.d/ssh` which tells to use `pam_ldap.so` for authentication of the user.

Other authentication modules in `/etc/pam.d/*`:

chfn - change user name and information  
chsh - change login shell  
cron - daemon to execute scheduled commands  
cupsys - printer system  
kde - x-windows. Graphical window system in Linux.  
kscreensaver - screensaver  
login - begin session on the system  
passwd - change user password  
ppp - point to point protocol transmission  
samba - Windows/Linux integration tool  
su - change user ID or become super-use  
other - Other programs that need to use PAM-authentication

The goal is that every service that provide user logins, authenticates the user and its password against the LDAP. The password should be sent encrypted over the network.

When working with Pykota as a tool for printing accounting, Postgresql is used to store information about users and quotas. For authenticating against this database a PAM module should be used. In addition to this it's also recommended that mailman and Webmin uses a PAM module for authenticating. None of these modules are currently realized in Skolelinux, but will hopefully come at a later time.

### **SNMP**

Simple Network Management Protocol (SNMP) has become the standard for network management. It is a simple solution, requiring little code to implement and vendors can easily build SNMP agents to their products. SNMP is extensible, allowing vendors to easily add network management functions to their existing products. Management in networks consist of three pieces<sup>[18]</sup>

1. A management Information Base (MIB) that specifies what variables the network elements maintain. I.e. the information that can be queried and set by the manager.

2. A set of common structures and an identification scheme used to reference the variables in the MIB. This is called the Structure of Management Information (SMI) [19].

3. The Protocol between the manager and the element, called the Simple Network Management Protocol (SNMP)[20].

SNMP is the network service that underlies many network management programs. SNMP was designed to be a consistent interface to gather data and set parameters for various network devices, which include switches and routers, network hosts running almost any operating system, printers, and more. In our context Pykota uses SNMP to ask printers for its page counter. This is used to read how many pages a printer has printed. For an SNMP manager to communicate with an agent, it must know what various data values a particular agent is monitoring. These data values and the contents associated with them are defined in the MIB.

The MIB is not a database; it doesn't actually hold any data itself. It's simply a definition of the data values that can be monitored or modified. These data definitions and naming conventions are used by the SNMP agent software, which collects the actual values for these definitions from the device. E.g. the MIB value for getting the pagecount from HP printers is 43.10.2.1.4.1.1. This is used by Pykota (See Increment 1 - printing quotas)

## 2.3 Increment 1 - Printing quotas

### Printing in a multi-user environment

The possibility of printing in a network is important. Sharing printers and using a printer from any host. Printing in a multi-user environment is all about getting the most out of a single printer across multiple machines.

Printing can be a problem area for system administration. A printer is often the only mechanical part of a network. Paper jams is inevitable. Then printing queues grow and jobs may be lost. When a paper jam occur it is most likely that the printer will be restarted. Then the buffer in the printer will be lost and users have to print the jobs one more time. This can also have an effect on quotas. Users may loss prints they never have had printed out. Other problems regarding printers are; toner and ink do get empty, mechanics do get worn out and break. This can all lead to administration problems and user confusion.

A printer in a multi-user environment is most often connected to a serial port, parallel port or as a host connected to a hub or a switch. Serial ports have the disadvantage of being very slow (serial port run at 38.4 Kbps as opposed to 150 Kbps for a parallel port). Firewire and USB are also possible but it is not as widespread and usual for network printers. So parallel or Ethernet connection is most often the solution. Most of the printers for networks today have both a parallel port connection and Ethernet connection. Some printers only have a parallel port. If an Ethernet connection is needed on these printers it's possible to use the Hewlett Packard's (HP) Jetdirect system[21]. This is a box that you can connect your parallel cable printer to and then use an ordinary Ethernet connection from this box. The advantage of using an Ethernet connection is obvious. E.g. in a computer lab situation , where an Ethernet is running trough the

building and the printer server is in another room than the computer lab. The printer don't need to be connected to the server directly, it just need be connected to the network. The printer is just another host on the network.

Having a quota system for printer use is common in a LAN situation. Especially since prints are expensive in a school setting. Quotas for each user limit the use of toner and paper. A disadvantage regarding printing quotas is that when users get a quota they may feel that they have to use all the prints they have been given. This psychological effect can lead to the opposite of saving prints. But quotas rule out the possibility for a single user to misuse the printers and override the budget. In a school setting a single user may override the paper budget for the entire school. So quotas are often the best and only solution in a school setting.

### **Printing in Linux**

Print setup and sharing under Linux opens for many choices. Most Linux systems today come with BSD printing subsystem as standard printing subsystem[22]. LPRng[23] is enhanced version of BSD printing subsystem. Both use the lpd printing daemon. CUPS (Common Unix Printing System) are a more powerful printing subsystem. This is the printing system that Skolelinux use. Other possibilities are to make Linux point to a remote printer on an AppleTalk, TCP/IP, or SMB network. This project does not cover printers on other networks than a Linux network.

A printer subsystem includes the following components[24]:

#### **-Printers**

Devices like a laser printers, inkjet printers etc. Attached to a local computer or as a stand-alone device in a network.

**-User commands to initiate printing**

The user specifies the file to print, witch device to print on and other necessary instructions. Called a print job.

**-Queues**

Queues store and sequentially process print jobs. It's a simple FIFO (First In First Out) queue.

**-Spooling directories**

Spooling directories hold pending jobs. Files may be copied to spooling directories before it is being send to the printer.

**-Server processes**

Server processes accept print requests. Then set up and store the print file and transfer the jobs from the spooling directory to the actual printer.

**-Filters**

Print filters are programs that convert from multiple file types to one that the printer understands. Runs automatically for each print job.

**-Administrative commands**

These commands stop and start the printing subsystem or specific printers and manage queues and individual print jobs. Configuration files are usually used to specify the various characteristics and desired settings for each printing device. They are most often modified automatically by the various administration tools/commands.

**-Remote printing**

Nowadays, remote printing is at least as common as local printing. The use of printer servers is common.



The clients send the print jobs to the printer server.  
The server then does the job of printing.

A prime consideration when printing a file, is whether its format is compatible with the printer. The wrong format can give a lot of useless pages. The file type and printer type must match perfectly to get the output just right, therefore filters are important. Print filters in most cases convert to either Postscript[25] or PCL. PCL (Printer Control Language) has become a standard for controlling laser and Inkjet printers. Postscript however is the most flexible, but not all printers support this method. Nearly all Linux software that produces printable output, produces it in Postscript.

The best choice for a network printer in a Linux environment is to buy a printer with Postscript support in firmware. Most laser printers today support Postscript, but outside the laser printer domain, Postscript support is scarce and it can be a costly add-on. Linux/Unix software, and the publishing industry in general, have standardized upon Postscript as the main printer control language. It's several reasons for this:

- Timing.** Postscript arrived as part of the Apple Laserwriter. These printers fit perfectly with Macintosh machines. And this system was largely responsible for the desktop publishing revolution of the 80s.

- Device independence.** Postscript programs can be run to generate output on almost any sort of printer mechanism without the original program needing to be changed. Postscript output will look the same on any Postscript device.

- It's a programming language.** Postscript is a complete programming language. Software can

be written in it to do almost anything.

**-It's open.** Although Adobe invented it and provides the dominant commercial implementation a lot of other vendors has independently coded implementations.

Ghostscript [26] is a program in GNU/Linux that can be used to print Postscript on non-Postscript printer. In Microsoft Windows Adobe PressReady work similar to Ghostscript in GNU/Linux.

Printing quotas is also of course possible in a GNU/Linux environment as most other modern network environments. LPRng support pr user printer quotas. PrintBill [27] is an administration tool for printer quotas using LPRng. CUPS also has quota system but this system has limitations. Printing quotas in CUPS is the main problem we have to solve in this increment.

### **How CUPS work**

The Common UNIX Printing System (CUPS)[28] is a cross-platform printing solution for all Unix environments. It is often used in networks where centralized printing administration is preferable. It is based on IPP (Internet Printing Protocol) [29] and provides printing services to most Postscript and PCL printers. CUPS is provided under the GNU General Public License[8] (GNU GPL), which shortly means that you can use and copy the software without charge.

CUPS was from the beginning designed to address printing within a networking environment, rather than being focused on printing within a single system. But it has features to support both local and remote printing, as well as printers directly attached to the network.

Internet Printing Protocol (IPP) [30] used by CUPS is supported by most current printer manufactures an operating systems. IPP is an upcoming standard for printing. Up to now printer vendors

had to support a wide range of protocols. IPP aims to solve this. It is implemented as a layer on top of HTTP (port 80), and through HTTPS (port 81) it includes support for security-related features like user authentication, access control, and encryption. This makes it possible to administrate the printer server from a "normal" client through the web browser. One drawback might be that the printer server systems require a web server in the network.

CUPS also supports multiple printer servers on the same network. In this case the different servers will tell the other servers on the system about their installed printers and all of the printers will be available for all clients. This makes it possible for all the thin clients on the 192.168.0.0/24 segment to reach a printer connected to the 10.0.2.0/23 segment in Skolelinux network and the other way around. This because the CUPS server on the LTSP-server (192.168.0.0/24 segment) communicates with the other CUPS servers on the 10.0.2.0/23 segment.

When starting a print job, CUPS separates the print job handling and device spooling functions into modules. Print jobs are given an identifier number and a set of associated attributes like destination, priority, media type and number of copies. Filters may be specified for print devices and print queues in order to process print jobs. This allows a user or application to print different types of files without extra effort. CUPS provides filters for printing many types of image files, HP-GL/2 files, PDF files, and text files. CUPS also supplies Postscript and image file Raster Image Processor ("RIP") filters that convert Postscript or image files into bitmaps that can be sent to a raster printer. When use of these filters, print jobs are filtered on the CUPS server before sending them to a printer. In this project we have used Pykota as a filter that is able to reject the printjobs (more about this in the Pykota chapter).

CUPS supports commands like `lpr`, `lpq`, `lprm`, `lp`, `lpstat` and

cancel, known from the BSD and LPRng systems. With the command `lpadmin -p printer -o job-page-limit=value` it is possible to set the page limit for per-user quotas on the specific printer. The value is the number of pages that can be printed.

CUPS also have the possible to set up printer classes. A class consists of many printers and a job sent to the class is automatically forwarded to the first available printer in the class.

### **Analysis concerning printer quotas**

The background for this increment is to give the system administrator control over the users print consumption. This is mainly useful for controlling students, but may be convenient for the teacher's accounts also. The system administrator should be able to set a maximum print limit for the students, whereas for the teachers it may be suitable to monitor their consumption. Eventually restrictions may be set afterward. As an extension to this, a useful feature is to control which printers different groups can send print jobs to. Students in example, may not be able to print to the printers station at the teachers staff room.

Skolelinux wish to store all information about the users and their access privileges in the LDAP-base. This should also include information about the users print account. It was a demand from Skolelinux to use CUPS as printing subsystem.

To solve this increment, we started early in February to scan the marked for possible solutions, and tried to come in touch with people who had experience in the field. We used Google for search and participated on mailing lists (Skolelinux, Openldap, CUPS) and Skolelinux Irc-chat. Unfortunately we did not find anybody who successfully had implemented CUPS with users in LDAP. We found many emails and questions about how to solve this, but very few answers. In other words it seems that this is a topical of interest, which many people wish to solve.

When looking for existing software to solve this issue, we found PrintBill [31], Printquota [32] and Pykota[33].

CUPS, which is a new generation of printing environment for UNIX like systems, features a simple print quota solution, but it only supports accounting per printer. You can set a maximum amount of prints on the specific printer, but then everyone printing on this printer will have the same amount of prints. E.g. if user `olem` has a print limit on 500 pages, the user `tc` will also have a print limit on 500 pages. As a result of this you can't set `tc`'s limit to 200 pages and still let `olem` print 500 pages, which may be preferable in a school context. CUPS lack a lot of features that is preferable in a larger network system. The CUPS quota system has small possibilities of configuration. If a user is empty of prints, it is not possible to give him more prints without giving every other user a new quota on the actual printer. This quota system becomes too weak in the Skolelinux relation.

CUPS store the quota information in the `/etc/cups/printers.conf` file.

Regarding PrintBill and Printquota, it does the accounting job fine but doesn't work with CUPS, only with LPRng. Because of the demand of using CUPS in Skolelinux this was not a satisfactory solution.

Then we looked closer into Pykota and found it rather interesting. It uses CUPS, the developer Jerome Alet, planned to implement an LDAP storage backend.

Unfortunately the to-do list of the Pykota project was in order of importance and looked like this:

- Finalize the implementation of group quotas.
- Group administrators (think `quotagrdmins` for disk quotas).
- Documentation...
- LDAP storage backend

Currently Pykota just supports a Postgresql storage backend.

### **Pykota as a technology**

Pykota is a complete print quota software solution for CUPS. It's Free Software distributed under the GNU General Public License[8]. It works by directly querying the printers for the number of pages it has printed. This information is stored in a Postgresql-database. You can set up user accounts where you specify each users soft limit and hard limit of prints. We have now installed version 1.02 which has features like:

- Per printer user quotas.
- Administrator, requester, grace delay, and access policy defined either globally or per printer.
- Automated email warning of users above quota to the user himself and/or to the print quota administrator. This can be disabled if needed.
- CUPS filter for quota accounting : `pykota`
- Command line print quota editor : `edpykota`
- Command line print quota report generator : `repykota`
- Command line print quota automated warning sender : `warnpykota`
- Command line tools mimic the well-known Linux disk quota utilities for easier mastering.
- CGI script to allow your users to access their print quota usage report remotely.
- Centralized storage of quotas : you can manage quotas for different printers on different print servers and store them all on the same quota storage server. (All your printers must have a unique name, but this may change in a future version.)
- SNMP querying of any networked SNMP-enabled printer.
- External command querying (requesters) of any printer : provided you've got a printer which can report its life time page counter, and a script to read it, the printer is supported.

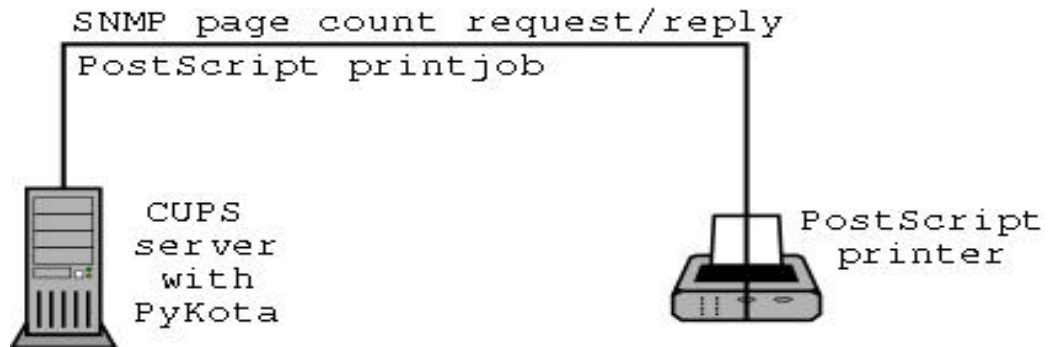


Fig 2.4 - CUPS/Pykota printer communication

How does Pykota act when user `tc` initializes a printjob?

Pykota get querying the printer via SNMP (see fig 2.4) for its total pages counter, just before the beginning of a job, and use this to modify `tc`'s quota. This means that you're always late of one print job, but this is generally not a problem because a check is also done to see if user `tc` is allowed or not to print.

A problem may theoretically arise in batches of successive print jobs by different users when there's no sleep time between two jobs. The used pages may theoretically be attributed to an incorrect user in the case that the printer is asked for its page counter at the beginning of a new job and before the end of the previous job. This depends on the printer speed and time between jobs. We have only tested Pykota successfully with a HP LaserJet 4200n, but the creator of Pykota, Jerome Alet, says he so far haven't seen any problem with moderately used printers. This will also depend on CUPS internal behavior. If CUPS doesn't begin to send a job to a printer before the previous one is completely printed, there will be no problem.

At the time there is some restrictions in use of Pykota:

- the printer must support postscript
- the printer should be a network printer because it is not preferable to have a CUPS daemon running on each host

- the printer must support SNMP
- it is not able to set quotas for groups. This is fixed in Pykota version 1.03.
- there is no communication with LDAP, all users must also have an entry in the Postgresql database. This can be done at the same time when registering the user in LDAP. So if you add a new printer and already have registered the users in LDAP, you have to run the command for user quotas for every user who is allowed to print on the new printer.
- Currently the Pykota filter is used at the very last stage of the print mechanism, just before the final data is sent to the printer, but it may eventually be used earlier with some modifications to CUPS's filtering configuration. This does not work yet, but this will open for non-postscript printers also.

Pykota is being used as a filter in each ppd file. The ppd files is located in the `/etc/cups/ppd/` directory. It's a ppd file for each printer installed using CUPS. In each of these files it is necessary to define the filter to use. In our situation this is the Pykota program. As for now (Pykota 1.0) it is only possible to use Pykota as a filter in a ppd file that is adapted for Postscript. I.e. a Postscript driver is installed for the printer. The latest version of Pykota (version 1.03) also supports Appeltalk. This filter (Pykota) is called each time a print job is sent to the printer by the CUPS server. When a print job is requested on a Pykota filtered printer, the filter will first check if the user has prints left in his/hers account in the Postgresql database. Then Pykota uses SNMP to query the printer about its total page counter and compare this with the users page count. Then Pykota either reject or send the file to the printer.

The quotas in Pykota are easily set using the `edpykota` command. To view the quotas the command is `repykota`. This list all the quotas set up.



### Our solution with Pykota

The restrictions of no LDAP-backend for the Pykota program have been a problem for us. The only storage backend that is possible with Pykota at this time, is a Postgresql database. We contacted Jerome Alete, the developer of Pykota, and he wrote that the LDAP backend is planned, but it is not a top priority. He was happy to conduct if we decided to try to make the LDAP storage backend. We discussed this in the group, but found out that the LDAP storage backend for Pykota was a to big task at this time of the project. It would take to much time to learn the Python programming language, which Pykota is programmed in. We contacted Petter Reinholdsen, one of the main developers at Skolelinux, and he wrote that a Postgresql database was needed in the Skolelinux distribution. A solution with quota information in this database was satisfactory at this time, until Pykota comes with a LDAP storage backend. So our solution is with the original Postgresql database.

The users in LDAP are not synchronized in any way with the users in the Postgresql database. We have not put time into doing this because of the planned LDAP backend for Pykota. Probably the best solution at this time is to use the `edpykota` and `repykota` command to administrate the print quotas. Alternatively the users with quotas can be inserted into the PostgreSQL when the users get inserted into the LDAP base through Webmin. This can maybe be a future project in Skolelinux.

The administration tools `edpykota` and `repykota` give an easy interface for administration.

Edpykota is used like this:

```
tjener:-# edpykota -P "printername" -S  
"softlimit" -H "hardlimit" "user1" "userN"
```

Printername is the same as the ppd filename for the installed printer. This printer must have the Pykota filter line in its ppd file. Softlimit describe the users maximum prints. It is

possible to override the `softlimit`, but only in special time periode. This time period is described as grace delay set in the `pykota.conf` file. It is default set to 7 days. `Hardlimit` describe another limit. This limit is impossible to override. With a `softlimit` lower than the `hardlimit` the user and administrator receive a mail when the `softlimit` is reached and then they know that they are close to overriding their print quota. Setting the quota to groups is also possible from Pykota version 1.03. This makes it easier to administrate the quota system. E.g. the administrator can set a quota for all users in a group much faster then setting quotas for each user individually. Example output from `repykota`:

```
tjener:~# repykota
*** Report for user quota on printer serverprinter
Pages grace time: 7days
User          used      soft      hard      grace  total
total
-----
morten        -         4         5         5         34
oledahl       +         3         1         3        DENY   11
tc            -         3         5         5         3
                                         Total:   48
                                         Real: 2514
```

The first column from left represents users. The second column if they are under or over quota. The third show the number of pages printed since their quota usage was reinitialized. The fourth their `softlimit`, and the fifth `hardlimit`. The sixth representing the grace date or deny if grace date is reached. The last column shows the total lifetime page count for each user. The two last lines represent total lifetime usage for the printer, first one as a sum of all users life time page counters, last one as the last value read from the printer's internal life time counter. The program `warnpykota` is another tool in Pykota. This program is used to send the emails about quotas to the administrators and the users.

We have successfully installed Pykota with the Postgresql database on the file server in our test network. The printers added on the cups server must have the Pykota filter line in the ppd file for each printer that need a quota system. This must be put in manually by now. When the cups server receives a print job Pykota do the job of accounting. Default policy must be set in the Pykota configuration file in `/etc/pykota.conf`. Default policy must either be default deny or allow. A default deny policy will deny all user to print if they don't have an account in the Postgresql database for the printer they try to print on. This applies to all printers with the Pykota filter line in their ppd file. Without the filter line deny settings must be set in CUPS (`etc/cups/printers.conf`).

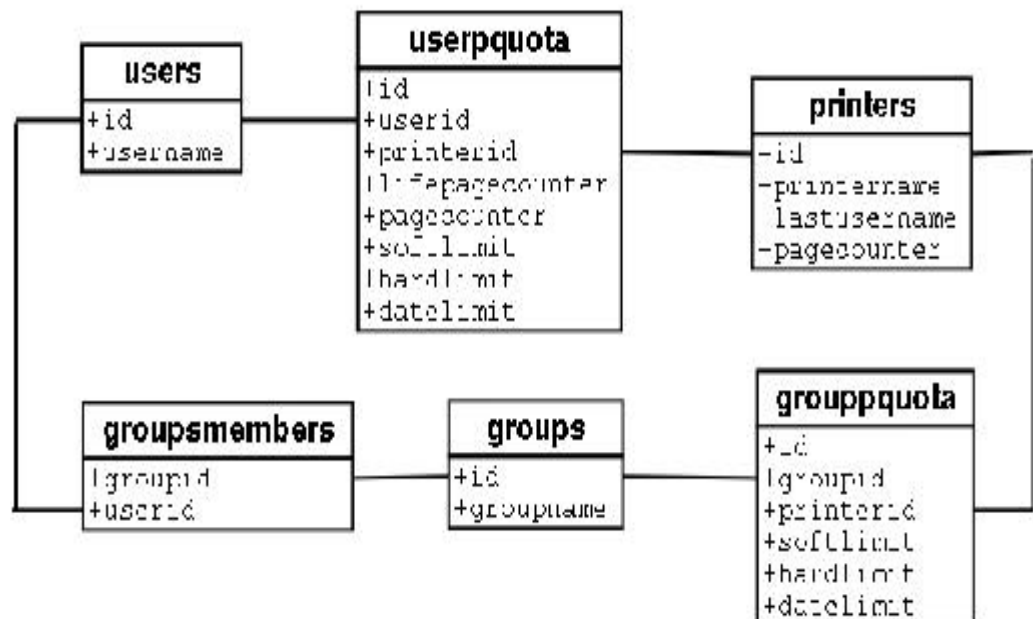


Fig 2.5 - Pykota database structure

The PostgreSQL database stores the account information. It has a structure (that stores information about groups, users, printers, see fig 2). A group is made up of several users, this function is supported by the `edpykota` command in the current version (1.03). The printer's page count is inserted in the printer's table

each time a printer job is sent, by use of SNMP. This pagecount is then used to update each users quota. Pykota now has all the information it needs to manage users quotas.

Pykota is currently only available as source code from CVS. Debian packages are not available, but it probably will be when Pykota is fully developed. The installation routines of Pykota 1.00 are attached as an appendix to our report. See appendix G.

## 2.4 Increment 2 - Netgroups in LDAP

### Netgroups in a multi-user environment

Netgroups are defined in a multi-user environment for administrative purposes. Netgroups is useful for defining lists of hosts, which belong to specific owners in a network situation. This can be used to limit access to different hosts for user groups. This is how it's intended in Skolelinux as limiting access to computers is needed. E.g. so that the students won't have access to the computers the teachers have in their staff room.

### Netgroups in Linux

Traditionally in GNU/Linux, NIS (Network Information Service) [34] is used to define netgroups. NIS is a kind of directory service developed by Sun Microsystems[35]. NIS was originally called YP (Yellow Pages) and was invented to simplify the administration of hosts and users originally defined in `/etc/passwd` and `/etc/hosts`. NIS was designed for a very open environment, in which significant trust among all systems is desired and demanded. NIS is often referred to as a security nightmare and is not recommendable in use. Because of the poor security features and security flaws in NIS, Skolelinux don't wish to use NIS.

To define a netgroup using NIS, `/etc/netgroup` is used. This file defines a netgroup including a (host, username, domain) tuple.

To avoid the security problems that NIS has, other or more future oriented directory services are required. NIS+ [34] is a newer development of NIS by Sun Microsystems[35]. This system has less security flaws than the original NIS, but some weaknesses still exist.

X.500 is another directory service that support netgroups, but is

very hard to implement (See directory service chapter). Then it's LDAP or OpenLDAP that we have tried to implement Netgroups into.

### **How NSS works**

Name Service Switch defines an interface between the core C library and a service module that implements a particular type of information storage. Applications under GNU/Linux interact with the system by calling functions of the C library.

NSS is designed to handle all the different directory services available. The `nsswitch.conf` (`etc/nsswitch.conf`) defines what directory or file you want to get database content from. E.g. `nsswitch` may have this entry:

```
passwd:          files ldap
```

When the system is trying to look up a password for a user it firstly will look in the files `/etc/passwd` and `/etc/shadow` then if it's not found here the password is looked up in LDAP directory.

### **Netgroups in LDAP**

The idea of making the netgroups work in LDAP is to make a replacement to NIS in LDAP. To use netgroups in ldap an `nss-ldap` module made by Padl software[36] is required. This module makes it possible to access the netgroup information in LDAP. The first version of this package that should support Netgroups is 2.04 (released in the early spring of 2003). The netgroup implementation in LDAP should work like it's described in fig 2.6.

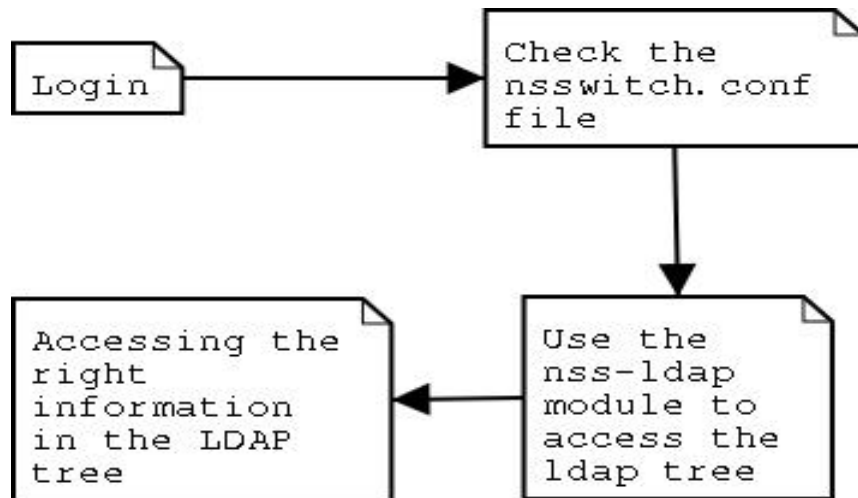


Fig 2.6 - NSS and LDAP

Configuration of the nss-ldap module is done by the file `/etc/libnss-ldap.conf` file. In this file i.e. the search entry for the netgroups in the LDAP base is defined.

### Our tests with netgroups

We have tried many different approaches to solving this problem. With some problems. The nss\_ldap modules we have tried, has lead to problems. The first nss-module we got to work was version 2.04-3(released as Debian unstable in may 2003).

RCF 2307 “An Approach for Using LDAP as a Network Information Service” [37], defines the function of netgroups. But previous versions (before 2.04) and the version in the Skolelinux distribution don't support netgroups.

We first thought the implementation of netgroups wasn't going to be a problem, but soon learned that it was. The netgroups weren't supported in the previous versions of nss-ldap although it was said to do so in the documentation. We got this confirmed both through testing, searching the Internet, other developers at Skolelinux and through communication with a system administrator at Borland.com (J. Corley).

Netgroup support in the `libnss-ldap` package was

implemented in version 2.04 this spring (2003). And is only available through SID (Debian Unstable Distribution). This package for NSS supports the LDAP server acting as a name service. This service can be configured through `/etc/libnss-ldap.conf` with e.g.:

- Where the LDAP server runs
- Top domain in the LDAP tree.
- Which LDAP version being used.
- Where to find the netgroups, groups and other information in the LDAP tree.

Another configuration file, which needs to be altered, is the configuration file for the Name Service Switch (NSS). This file can be found in `/etc/nsswitch.conf`.

This is where one specifies where NSS should find the netgroup (e.g `compat, files, ldap.`)

Then the LDAP needs to be given the information about the netgroup. This is done by `ldifs`, containing the user and machine information.

To test netgroups we made the file `/etc/netgroup` which is a file being used by NIS, but will here be used for testing if the netgroup commando (the netgroup commando is a program from the deb package `ng-utils` made by Petter Reinholdtsen, for testing netgroups) works. `apt-get install ng-utils`, if you have the Skolelinux source in `/etc/apt/sources.list`).

Into this file we added some test data:

```
allhosts (host,-, )
allusers (-,someuser,)
supergroup allhosts allusers
```

Output from the commando `netgroup allhosts`:



```
host
```

This shows the netgroup commando works fine.

Made a ldif from this test with:

```
perl -I /etc/migrationtools/  
/usr/share/migrationtools/migrate_netgroup.  
pl /etc/netgroup (by Luke Howard)
```

This returns the possible ldif from the testing in the  
/etc/netgroup file :

```
dn: cn=allhosts,ou=Netgroup,dc=padl,dc=com  
objectClass: nisNetgroup  
objectClass: top  
cn: allhosts  
nisNetgroupTriple: (host,-,  
memberNisNetgroup: )
```

```
dn: cn=allusers,ou=Netgroup,dc=padl,dc=com  
objectClass: nisNetgroup  
objectClass: top  
cn: allusers  
nisNetgroupTriple: (-,someuser,)
```

```
dn:  
cn=supergroup,ou=Netgroup,dc=padl,dc=com  
objectClass: nisNetgroup  
objectClass: top  
cn: supergroup  
memberNisNetgroup: allhosts  
memberNisNetgroup: allusers
```

This show how nss wants to have the netgroups defined in the LDAP database. The `nisNetgroupTriple` is build up just like the original NIS tuple syntax, i.e. (machine, user, domain).

### **Our installation routines**

Install the new `libnss-ldap` module from SID, with

netgroup support.

```
dpkg --install libnss-ldap204.deb
```

Edit /etc/nsswitch.conf with netgroup: ldap instead of netgroup: files. Or netgroup: files ldap if both the etc/netgroup file and ldap is in use.

Add the ldif files with netgroup information to the LDAP tree.  
Example from our test implementation:

#### Machinegroup

```
dn:  
cn=allhosts,ou=Netgroup,dc=skole,dc=skolelinux,dc=no  
objectClass: nisNetgroup  
objectClass: top  
cn: allhosts  
nisNetgroupTriple: (dhcp-003,-,)
```

#### Usergroup

```
dn:  
cn=allusers,ou=Netgroup,dc=skole,dc=skolelinux,dc=no  
objectClass: nisNetgroup  
objectClass: top  
cn: allusers  
nisNetgroupTriple: (-,tc,)
```

#### Relationship between allhosts and allusers

```
dn:  
cn=supergroup,ou=Netgroup,dc=skole,dc=skolelinux,dc=no  
objectClass: nisNetgroup  
objectClass: top  
cn: supergroup  
memberNisNetgroup: allhosts  
memberNisNetgroup: allusers
```

This is added to the LDAP database with the command:

```
ldapadd -h 10.0.2.2 -x -D
cn=admin,ou=People,dc=skole,dc=skolelinux,dc=no -W -f TestNetgroup.ldif
```

The base string is not where to add the information in the LDAP tree, but for which user you are changing the LDAP tree with.

Here: admin.

We also had to change the configuration of the nss-ldap module in `/etc/libnss-ldap.conf`.

Our settings in `libnss-ldap` look like this:

```
host 127.0.0.1
base dc=skole,dc=skolelinux,dc=no
nss_base_group
ou=Group,dc=skole,dc=skolelinux,dc=no
nss_base_netgroup
ou=Netgroup,dc=skole,dc=skolelinux,dc=no
nss_base_passwd
ou=People,dc=skole,dc=skolelinux,dc=no
```

First we had to set which host the LDAP server is running on, in our case localhost using loopback interface (127.0.0.1). Then we had to specify where the `dn` of the search base of the LDAP tree. This is used as default search scope if nss doesn't find the entry. In our context searching from the top of the LDAP tree (See fig 2.1). `nss_base_group` is set to where the groups are defined in the LDAP tree. Then where the netgroups are defined in the LDAP tree. At last where the user information is found with `nss_base_passwd`.

Then it's possible to view the defined netgroups in LDAP using the `netgroup` command.

```
netgroup supergroup gives the output:
tc
dchp-003
```

Now the netgroups are defined and function the same way as it

had done in NIS, however now LDAP is in use.

## 3 CLOSURE

### *3.1 Discussion of the project*

#### **Development model**

Choosing a development model was one of the first things we had to do. Since the tasks from Skolelinux were split up in several increments, we soon decided that the incremental development model was a natural choice according to what we have learned in the system-engineering course.

But it soon became clear that the Skolelinux project works quite different than anything we have learned at school. Therefore we needed to sit down and learn about the open source development models, as this is what Skolelinux is about.

#### **Gantt chart**

One of the other things we initially did, was to set up a schedule with the Gantt chart, see `appendix A`. This has to some degree been followed.

Since the increment we first started to read about and prepare for was solved by Andreas Dahl at Linpro, it's obvious that it couldn't slavish be succeeded.

At the end of this project we have set up a Gantt chart as it looks according to what we have done and when we did it, see `appendix A`. Even though there have been some changes, the main completion phases are somewhat correct.

Comments on the Gantt chart:

- Pre-engineering: This section went according to the original Gantt chart.
- Research: It wasn't easy to see all the technologies needed to be learned when we first started. New technologies have been

introduced as we dug deeper into the subjects. Therefore the research has been going on continuously, parallel to the rest of the work and not on the 10 days as we originally planned.

•Main section: The LDAP programming, the 3 increments, have been changed to only two as described before. One group member, Trond Christian, got sick in the last week of this section. Before Trond Christian got the chickenpox we planned to do more on the netgroup problem, but this then changed to do the report writing.

No Norwegian user manual is needed. This was a misunderstanding. The group from NITH who's been working on user administration is the ones to make this.

•Project closure: This element is of course not done yet. But we should be able to make it in time.

### **Milestones**

At the start of the project we created 7 milestones.

Comments on the milestones:

•Project start, 06.01.03: The project started January 7<sup>th</sup>.

•Hand in the pre-engineering project: Was delivered in time.

•Finish the requirement specifications: This milestone was completed in time but has been modified during the project, as we've been talking to other developers in the project at the gatherings (see appendix F) and through the mailing lists.

•Finish the programming: This project has not been all about programming, but configuring, testing and a lot of research. We therefore decided for a better name and renamed this milestone to finish the implementation.

•Finish the Norwegian user manual: A milestone that was a

mistake and applies to the project group at NITH.

•Complete the main project report and present the project:  
These two last milestones are definite to the project, and are  
still to be reached.

### **3.2 Criticism of the project**

We were never given a distinct task description from Skolelinux, but had to make it mostly by ourselves. According to the subjects, and what we thought were needed. This caused the project to start off a bit slow, since we had to read about more subjects than we otherwise would have had to. We have also spent quite some time trying to define this final specification. If we'd been given a more distinct document at the start of the project, things could perhaps have been done sooner. But again, this how system administration is being done.

Our problems in cooperating with the student group at NITH, might also be due to the task description. Since none of us really knew what should be done, and whose responsibility it was. Different deadlines didn't help the situation either.

The way things are being done in Skolelinux, is called 'gjørokratiet'. This way of organizing (the one who does something, decides), might not be the best suited for student projects with its strict time frames.

It took time to establish the right contacts with the other developers. Who to ask the right questions? We participated at gatherings, wrote mail on the Skolelinux mailing list and chatted at the Skolelinux irc channel (irc.ifi.uio.no channel #skolelinux). At the end of this project we started using the irc more frequently. This should have been done sooner, as we learned this was perhaps the best way of getting help from the right people for our project.

We feel that the gathering weren't as useful as they could have been. Lack of time and absence of developers with the right technological knowledge for our assignment, contributed to this. But we learned a lot of other things, with relations to system administration an open source development.



Trond Christian got the chickenpox at the end of this project. It put some stress on the rest of the group, but worked out anyway. This isn't exactly criticism, just bad luck.

### **3.3 What we have learned**

#### **Technologies**

To understand Skolelinux we had to acquire knowledge about many technical subjects. This includes technologies described in the two increments, and in the appendix containing different technologies (appendix E). Since all three of us are in the branch of study system administration, we think technologies like LDAP, PAM and printing are definitive matters we will be fronted in future work. That was also some of the reason why we started this project in the first place.

#### **Human aspects of working in a group**

When working three persons in a project group like this, we have learned the value of letting everybody state their opinion. It has been helpful in both understanding the problems and making solutions to them. Even though we haven't always agreed, we have let the majority rule. This has worked fine.

Some other thing we also experienced, and that we kind of knew from before, is that everything tends to take more time than one would expect. This might have helped us in understanding the value of setting off enough time at the end of the scheme, for future projects.

We have also learned how important it is to organize the group member's work, and that is why we each Monday meeting have set individual tasks. See appendix B for meeting reports.

### 3.4 Further work

Much of the work being done in open source developing is continuous and never finishes. This is also currently so for the two increments we have been working on and for the whole of Skolelinux

#### •Printing increment:

The creator of Pykota has a to do list which include implementing features such as: group administrators and LDAP storage backend. During our project he has produced several versions, and he has been very helpful in answering both installation- and other questions. He is also going to make a LDAP storage backend.

What could continue this project in direct tasks, is to make a GUI for user administration i.e. netgroups and printer accounting administration.

Make a Debian package and implement it to the Skolelinux distribution.

#### •Netgroup increment:

The support for netgroups in libnss-ldap didn't even come until this spring, and is yet to be fully tested.

A GUI for editing netgroups also needs to be made. This could very well be a webmin module for editing netgroups with machines.

Testing if netgroups work in `/etc/exports`. So that the administrator can export users home directories to `@allclients`, to limit who can use NFS.

Both of these increments are needed in Skolelinux. Netgroups

has a critical factor of high in Bugzilla, and needs to be fully implemented before a version 1.0 of Skolelinux can be published. Therefore any work that has been done in Skolelinux needs to be documented in English, so that others can use any experience we and the other student groups have dedicated us. This is the reason why Skolelinux insists on saving all student work, including the webpage, in CVS.

### ***3.5 Evaluation of the project group***

#### **Introduction**

The project group has worked well together. Ole Martin and Trond Christian have worked together in projects before this main project. These collaborations have given good results. Morten was new in the group, but we soon got to know each other. The collaboration has worked well.

From the start we all were highly motivated to get the most out of this project. We have been very eager to learn more about GNU/Linux, user- and system administration. The last year of our bachelor of computer engineering study, we have all chosen the system administration direction and we have taken mostly the same subjects.

#### **Organization**

The main project group has had an informal organization. This was natural because we knew each other, and knew what had to be done. We have successfully worked without a project leader. We have tried to motivate each other in hard periods. The only strict organization we have had, is that we have worked at least from 09:00 to 15:00 every weekday when it's possible. The only allowed absence has been lectures and sickness. This has been kept almost without exceptions and is well exceeded.

We all conducted equally in meetings and development seminars with outside sources.

We have had no problems with our informal organization.

#### **Distribution of the work**

The informal organization hasn't affected the project in any way. We have tried to divide the tasks after wishes and competence.

But many of the tasks have proved to be too difficult to solve alone. This has lead to almost all tasks have been worked on by all the group members. This has compromised the efficiency and productivity, but we feel that all the group members have learned equally much.

The workload of each group member has been almost alike. There are some differences in the amount of hours spend on the project. This is mainly because Trond Christian was away in the last week in April and the two first weeks of May because of chickenpox, when we worked a lot on this project report. Because of Trond Christian's chickenpox and that we have had individual assignments, some work might not be documented as well as it should be.

### **Project as work method**

This is our first big project work. Projects we have worked on before have been much smaller and less demanding than this project. Previous projects have given us some experience for this main project, but we have learned that a main project demand a lot more in planning, pre-engineering and structure.

Taking consideration to people when they are having a bad day or is tired, is easier to experience in large projects like this one.

This project has set high demands to each group member on what to do. Sometimes to great demands. It has not been possible for a single group member to solve things on their own. Collaboration with all the members and people outside the group has been essential. This has been done through mailing lists and development gatherings, see appendix F.

The project has been spared of major internal conflicts. Motivating for further work has been the biggest problem. a lot of problems have stopped the progress in work and this has lead to much confusion in the group. Some disagreements about

what to do and work methods has occurred, but have been solved along the way.

### **3.6 Conclusion**

We have been working on two different assignments in this main project:

#### **Increment 1 – printing quotas:**

The system we have chosen gives a satisfactory solution to the printing problems in Skolelinux. The developer of Pykota updates the software frequently, and has done so several times during this project. The system works well and has a lot of features.

#### **Increment 2 – netgroups in LDAP:**

Definition of netgroups is successfully implemented into the LDAP-base, but we haven't come so far as to limiting access to computers through the use of netgroups. This primarily because the libnss-ldap module didn't work as the RFC 2307 defined, before late in the project period.

The research we have done with netgroups in LDAP, can be helpful for future work with this problem.

Linux is a complex operating system, and it took a lot of time to understand it in the way we needed before solving the different problems. A well-formed problem description might have given us a faster introduction to the real problems.

The project has given us a lot of knowledge and experience in GNU/Linux, system administration, different technologies and working in a project group. We are sure that this will be useful for our future. All of us will take a master degree during the next two years, and will probably use the expertise, which we have acquired.



## 4 References:

- [1] Skolelinux's homepage, <http://www.skolelinux.no>
- [2] Linuxiskolen's homepage, <http://www.linuxiskolen.no>
- [3] OpenOffice.org's homepage, <http://www.openoffice.org>
- [4] Open source development page, <http://www.opensource.org>
- [5] The Debian Free Software Guidelines,  
[http://www.debian.org/social\\_contract](http://www.debian.org/social_contract)
- [6] The Open Source Defenition,  
<http://www.opensource.org/docs/definition.php>
- [7] Bugzilla homepage,  
<http://www.bugzilla.org/about.html>
- [8] GNU General Public License,  
<http://www.gnu.org/copyleft/gpl.html>
- [9] PADL Software Pty Ltd webpage, <http://www.padl.com>
- [10] Microsoft corporation. "Understandig the role of Directory Services versus RelationalDatabases.",  
<http://www.microsoft.com/windows2000/techinfo/howitworks/activedirectory/dsvsrd.asp>
- [11] Understanding X-500 - The Directory,  
<http://www.isi.salford.ac.uk/staff/dwc/Version.Web/Contents.htm>
- [12] Open LDAP Project Overview,  
<http://www.openldap.org/project/>
- [13] OpenSSL project homepage, <Http://www.openssl.org>
- [14] SASL homepage, <http://asg.web.cmu.edu/sasl/sasl-library.html>

- [15] Information about PAM,  
<http://www.kernel.org/pub/linux/libs/pam/whatispam.html>
- [16] User Authentication HOWTO,  
<http://www.tldp.org/HOWTO/User-Authentication-HOWTO/x101.html>
- [17] PADL Software Pty Ldy,  
[http://www.padl.com/OSS/pam\\_ldap.html](http://www.padl.com/OSS/pam_ldap.html)
- [18] W. Richard Stevens. "TCP/IP Illustrated, Volum 1. Addison-Wesley", Chapter 25 SNMP. 1994.
- [19] "RFC 1155",  
<http://www.faqs.org/rfcs/rfc1155.html>. 1990
- [20] "RFC 1157",  
<http://www.faqs.org/rfcs/rfc1157.html>. 1990
- [21] Grant Taylor, "The Printing HOWTO, version 5.15",  
<http://www.linuxprinting.org/howto>. 2001
- [22] Aileen Frisch. "Essensial System Administration", O'Reilly & Associates. Chapter 13 Printers and the Spooling Subsystem, 2002
- [23] Patrick A Powell, "The Printing Cookbook",  
<http://www.lprng.com/PrintingCookbook/index.html>, 2001
- [24] Aileen Frisch. Essensial System Administration. O'Reilly & Associates. Chapter 13 Printers and the Spooling Subsystem, 2002
- [25] Grant Taylor. "The Printing HOWTO version 5.15",  
<http://www.linuxprinting.org/howto> 2001
- [26] Ghostscript Home Page. <http://www.cs.wisc.edu/~ghost/>
- [27] About Debian Printbill package,  
<http://packages.debian.org/unstable/text/printbill.html>
- [28] CUPS Homepage, <http://www.cups.org>
- [29] "RFC 2567",  
<http://www.networksorcery.com/enp/rfc/>

rfc2567.txt. 1999

- [30] "RFC 3510", <http://www.faqs.org/rfcs/rfc3510.html>. 2003
- [31] Printbill homepage, <http://ieee.uow.edu.au/~daniel/software/printbill/>
- [32] Printquota homepage,  
<http://printquota.sourceforge.net/>
- [33] Pykota homepage,  
[http://www.librelogiciel.com/software/PyKota/action\\_Presentation](http://www.librelogiciel.com/software/PyKota/action_Presentation)
- [34] Hal Stern, Mike Eisler and Ricardo Labiaga,  
"Managing NFS and NIS", O'Reilly & Associates. 2001
- [35] Sun Microsystems homepage, <http://www.sun.com/>
- [36] PADL Software Pty Ltd, <http://www.padl.com>
- [37] "RFC2307, An Approach for Using LDAP as a Network Information Service"

## **5 Appendices**

***A - Gantt charts***

***B - Meeting reports***

***C - Status meeting reports***

***D - Log of hours used***

***E - Technologies***

***F - The Gatherings***

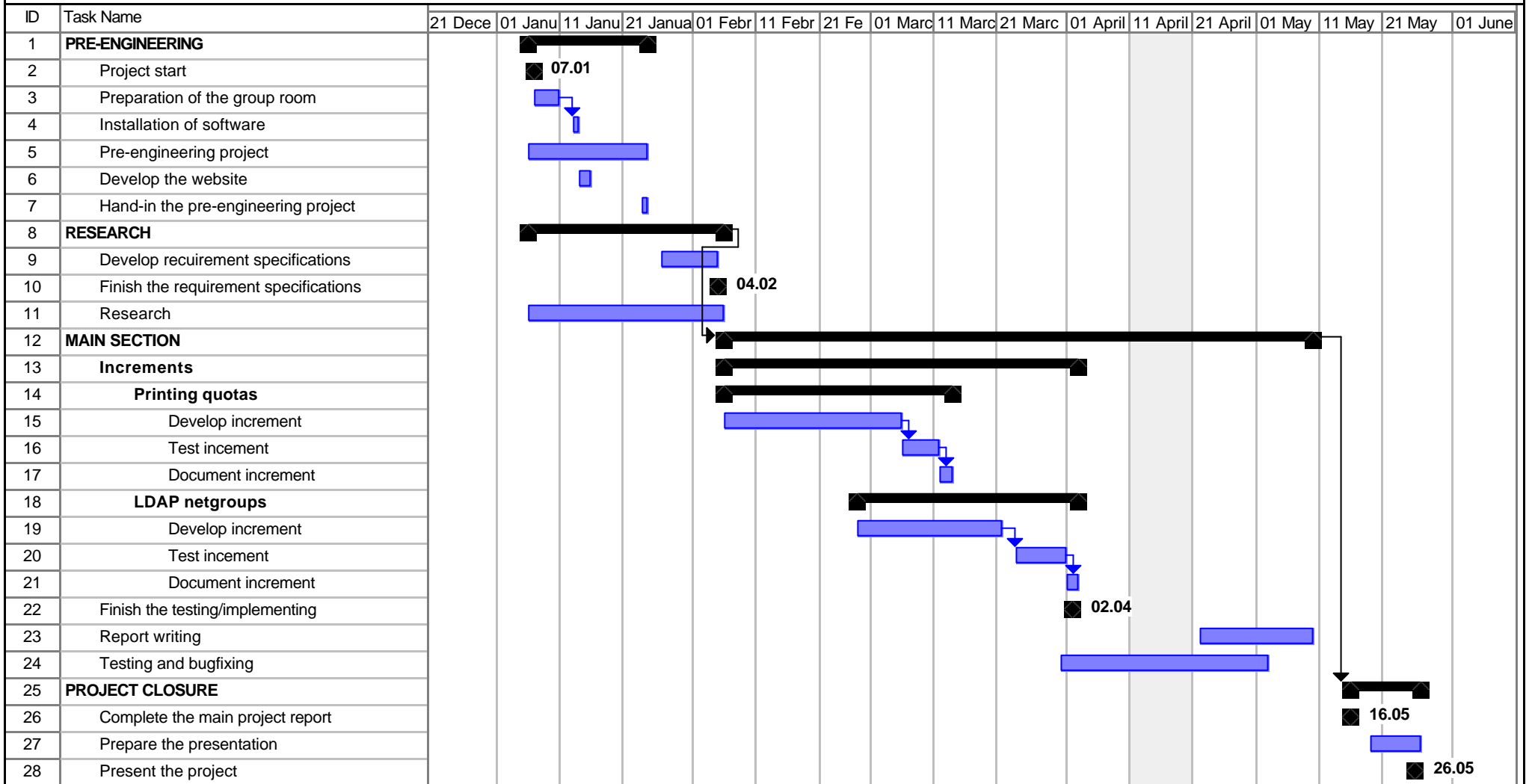
***G - Pykota installation guide***

***H - Pre-engineering project***

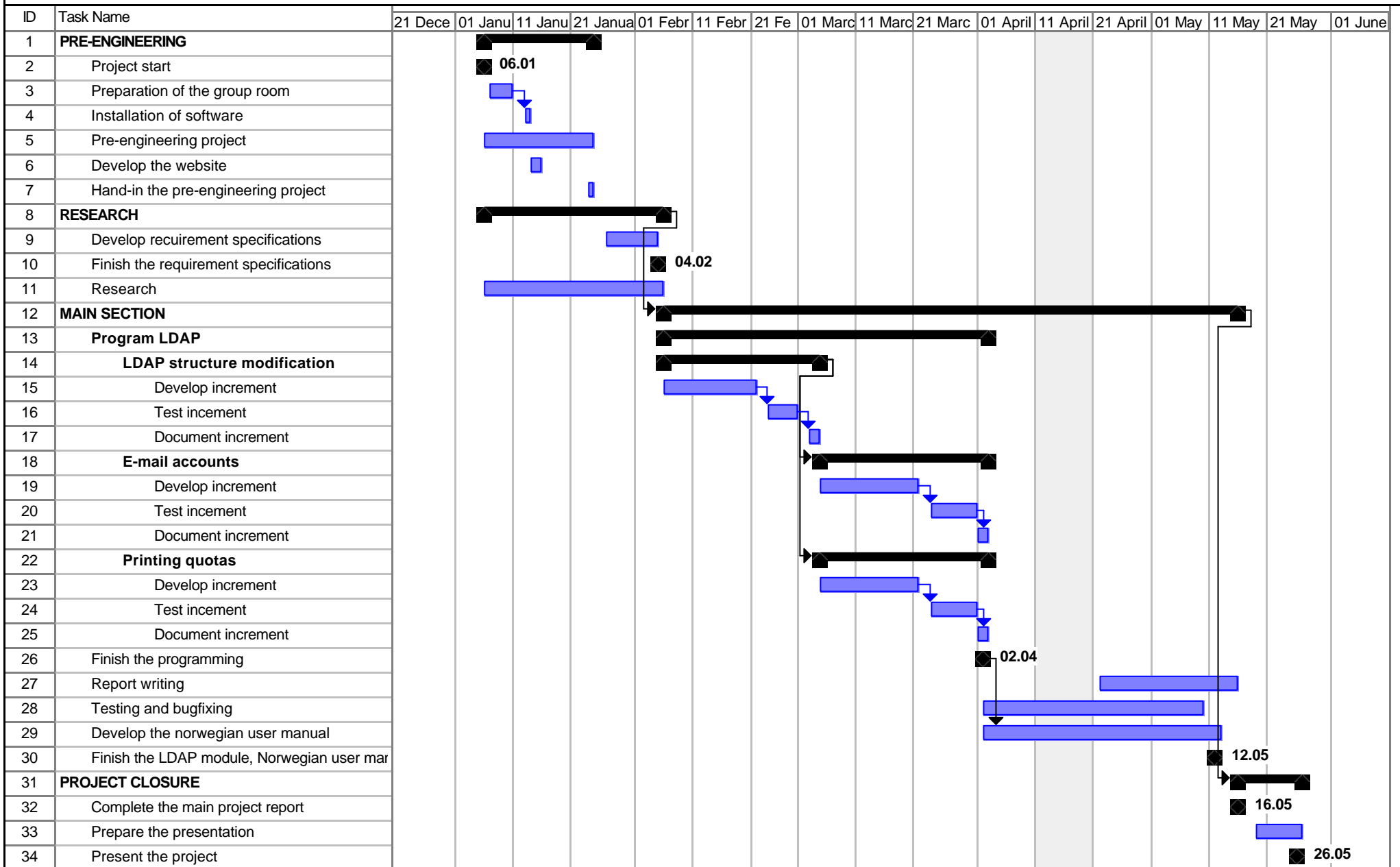
***I - Project contract***

***J - CD-ROM***

## Final Gantt chart



# Original Gantt Chart



# MØTERAPPORT 1

**Tidspunkt:** 27. januar 2003, 09.30

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

## **1. Status**

Forprosjekt og webside er ferdig. Kontrakt er underveis.

## **2. Ukens oppgaver**

Installere skolelinux testmiljø.

Planlegge utviklersamling, 31.jan – 2.feb.

Forberede litt kravspek, før utviklersamling.

Mounte prosjektområde/web.

Lese LDAP.

## MØTERAPPORT 2

**Tidspunkt:** 3. februar 2003, 10.00

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

### 1. Status

Kontrakt er levert.

Lest LDAP.

Nettverket er oppe og går, mangler tynnklient.

Utviklerseminar på Ulsrud Videregående Skole gikk bra. Lært CVS og fikk litt mer innblikk i hva vi skal produsere. Fikk kontakt med Andreas Dahl som har og arbeider med LDAP delen. Ser at det blir vanskelig å følge utviklingsmodellen vi har satt opp i forprosjektrapporten. Målet vil forandre seg og Gantt skjemaet må justeres.

Fant ut at kravspeken må ferdigstilles senere i prosjektperioden. Må jobbes med underveis. Frode Jemtland foreslo at vi skulle jobbe med å lære oss nødvendige ting i hele Februar, implementer i Mars og teste og gjøre oss ferdige i April. Noe som vi ser at er fornuftig. Det fører til forandringer i gantt skjemaet.

Ennå ikke fått mountet med SMB, skal løses i dag.

### 2. Ukens oppgaver

Endre Gantt skjema.

Få opp tynnklient.

Trenger å lese om NSS og PAM. Finne og skrive ut fakta.

Mounte prosjektområde/web.

Lese LDAP.



## **MØTERAPPORT 3**

**Tidspunkt:** 10. februar 2003, 10:30

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Fikk beskjed torsdag (6. feb) ettermiddag om ny utviklersamling i Stavanger helga 21-23 februar. Vi måtte bestille flybiletter fredag 7. februar, med andre ord ganske kort frist...

På denne samlingen har vi fått i oppgave å holde kurs i hvordan installere Skolelinux for lærere som skal jobbe med IT-drift.

Fant forrige uke en bra "HOWTO" som omhandler PAM mot LDAP.

Har mountet felles prosjektområde og www-område mot vår egen skolelinuxtjener.

Har fått tynnklienten til å fungere.

### **2. Ukens oppgaver**

Dokumentere utviklingsmodellen vår, open source (Ole Martin).

Lage oversikt over tjenester som trenger autentisering mot LDAP (TC).

Oppdattere Gantt-skjema (Morten).

Forberede kurs under utviklersamling 21-23. februar (Morten).

Fortsette lesinga om LDAP og PAM.

# MØTERAPPORT 4

**Tidspunkt:** 18. februar 2003, 10:30

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

## 1. Status

Utviklingsmodellen er nesten ferdigdokumentert.

Ganttskjema er oppdatert.

Vi har laget skjermbilder av installasjonen til Skolelinux som forberedelse til undervisningen på utviklerseminaret.

Bestemt oss for å satse på å få til oppgaven med CUPS godt, istedenfor å dra inn netgrupper midt i.

Eventuell utvidelse av oppgaven: Kryptering av passord og brukernavn over lokalnettet.

Oppsettet for `/etc/pam.d/other` autentiserer ikke alle tjenester på den sikrest måten (ikke satt opp med paranoid oppsett. Vi skal forsøke å teste om dette fungerer på den sikreste måten

## 2. Ukens oppgaver

Finne dokumentasjon om CUPS. (CUPS mot LDAP i [v.1.2??](#)). Lese CUPS.

Sette opp skriver i lokalnettet med CUPS i Skolelinux.

Alle gruppens medlemmer møter på utviklerseminar i Stavanger til helgen.

## **MØTERAPPORT 5**

**Tidspunkt:** 24. februar 2003, 10.30

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Skriver fungerer på nettet vårt.

Gantt skjema er oppdatert.

Ferdig med utviklerseminar i Time kommune. Hold kurs for it-veiledere i grunnskolen fra Stavanger området. Kurset var en suksess. Vi lager et vedlegg i hovedprosjektet der vi skriver om erfaringer fra kurset og legger ved kursmaterialet.

Fått kontakt med Jakob Breivik Grimstveit som kunne mye om LDAP. Var villig til å hjelpe til med problemer over mail.

### **2. Ukens oppgaver**

Skrive ferdig vedlegg på en Engelsk. Med kurs materialet(norsk) og erfaringer fra kurset. (Morten).

Lese og forstå CUPS.

Se om det bør lages printekvote felt i LDAPen og evt. hvordan. (TC).

Se på PyKota som skal støtte printerkvoter i cups(pr bruker).(Ole Martin).

Kravspek. jobbes med på tirsdag av alle 3. Kl 9 -16..

## **MØTERAPPORT 6**

**Tidspunkt:** 3. mars 2003, 09:45

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug

**Ikke tilstede:**  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Har funnet ut at man kan legge til og inkrementere/dekrementere ett felt i LDAP-treet. Dette er aktuelt i forbindelse med printerquota.

Installasjonen av PyKota feiler (prosessen henger). Vi vurderer å droppe PyKota, dette er siste uke med testing. Ole Martin følger med om det kommer en ny versjon før vi er ferdige med prosjektet.

Ferdig med første status rapport.

Godt i gang med kravspesifikasjonsdokumentet.

### **2. Ukens oppgaver**

Finne ut hvordan CUPS registrerer utskriftene, og om det er mulig å hente ut denne informasjonen, evt. hvordan. (Ole Martin)

Teste PyKota på reinstallert ("rein") maskin. (Ole Martin)

Lage felt for printerkvote i LDAP-treet. (TC)

Starte med inkrement 2: Finne ut om nettgrupper --> Network Service Switch (NSS). (Morten)

Kravspek gjøres ferdig av Ole Martin og TC på tirsdag.

# MØTERAPPORT 7

**Tidspunkt:** 10.mars 2003, 10:00

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

## 1. Status

Petter Reinholdsens i skolelinux har laget et c program som overlater printerkvoter til andre program enn CUPS. Kan være en ide for oss. Må finne/lage et program som lagrer kvotene. Petter Reinholdsens mener også at en PostgreSQL database må inn i skolelinux uansett, så PyKota kan nok brukes slik den er.

Ny versjon av PyKota er kommet på CVS.

PyKota fungerte i forrige uke, men har problemer med å registrere brukere i PostgreSQL.

Eget felt i LDAP treet er under vurdering, se om vi trenger det. Hvis vi trenger det må vi legge det i inetOrgPerson eller posixAccount objektklassen i LDAP.

Må antagelig lære oss Python.

Nettgrupper er vanskelig å løse i NSS/LDAP. Morten har sendt mail som vi venter svar på. Mailet med en i Borland.com som ikke har greid å implementere dette med LDAP.

Kravspek ikke helt ferdig ennå.

## 2. Ukens oppgaver

Rangere funksjonelle krav, finne ut hva vi skal satse på (Alle).

Jobber videre med nettgrupper (Morten).

Få PyKota til å fungere med PostgreSQL(Ole Martin).

Mail til NITH gruppe om dem planlegger å lage script for masseinnleggelse av brukere (Morten).

Synkro mellom PostgreSQL og LDAP (TC).

Sette opp liste over ting vi skal skrive om (Alle).

## **MØTERAPPORT 8**

**Tidspunkt:** 17. mars 2003, 09:45

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Satser på pykota. Må justere litt for å få det til å fungere.

Venter på mail fra Dagfinn Ilmari Mannsåker som hadde løst netgrupper. Morten jobber videre med å få tak i han.

Morten har skrevet om DHCP og CVS. Vi fortsetter på lista denne uka.

Har fått tilbakemelding fra NITH om masseinnleggelse av brukere. De vil at vi skal gjøre en oppgave der, vi vurderer dette til neste uke.

### **2. Ukens oppgaver**

Jobber videre med nettgrupper (Morten).

Få PyKota til å fungere med PostgreSQL (Ole Martin og TC).

Mail til NITH gruppe om masseinnleggelse av brukere (Morten).

Skrive om diverse teknologier til rapport (Alle).

## **MØTERAPPORT 9**

**Tidspunkt:** 24. mars 2003, 09:30

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Må finne ut hva vi skal satse på. PyKota virker ikke, selv om vi har hatt endel direkte kontakt med utvikleren.

Vi har fått en kort hjelp av Dagfinn Ilmari Mannsåker på hvordan sette opp nettgrupper. Skal prøve dette denne uka.

Det ser ut som om det forventes at vi jobber med masseinnleggelse av brukere på mailinglista til Skolelinux. Vi skal prøve å få til et møte med Erik om dette på tirsdag.

### **2. Ukens oppgaver**

Jobber videre med nettgrupper (Morten).

Siste uka med å få PyKota til å fungere (filteret funker ikke i bruk). Skrive om erfaringene på hvorfor Pykota ikke funker slik den skal (Ole Martin og TC).

Skrive om div teknologier til rapport (Alle).

Sette opp mal for rapport (Ole Martin).

# MØTERAPPORT 10

**Tidspunkt:** 07. april 2003, 09:00

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

## 1. Status

Delvis ferdig med pykota. Fått oppsett på filtjeneren til å fungere. Begynt å dokumentere inkrementet.

Ikke fått tak i Dagfinn Ilmari Mannsåker på hvordan sette opp nettgrupper. Har problemer med å få kontakt med LDAP servern. Modulen som skal gi netgrupper er installert (libnss-LDAP 204.2).

Vi satser på å løse nss\_LDAP og videre jobb på PyKota.

## 2. Ukens oppgaver

Levere og sette sammen delvis prosjektrapport til Erik Hjelmås på torsdag.

Skrive ferdig increment 1 (Ole Martin).

NSS\_LDAP (Morten og TC).



# MØTERAPPORT 11

**Tidspunkt:** 22. april 2003, 09:45

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild

**Agenda:** 1. Status  
2. Ukens oppgaver

## 1. Status

Levert delvis prosjekt til Erik Hjelmås. Han var fornøyd med det vi hadde skrevet. Kom med tips på hvordan vi bør fortsette å skrive.

Fortsatt ikke løst netgruppe problemet.

Har delvis ferdig en innholdsfortegnelse.

## 2. Ukens oppgaver

Skrive ferdig inkrement 1 (Ole Martin).

NSS\_LDAP (Morten og TC).

## **MØTERAPPORT 12**

**Tidspunkt:** 28. april 2003, 09:30

**Tilstede:** Ole Martin Dahl  
Morten Sporild

**Fraværende:**  
Trond Christian Frøhaug (Syk)

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Inkrement 1 en er forholdsvis ferdig skrevet.

Får ikke nss\_ldap til å fungere.

Trond Christian har fått vannkopper. Kan bli lenge syk.

### **2. Ukens oppgaver**

Begynner å dokumentere inkrement 2.

## **MØTERAPPORT 13**

**Tidspunkt:** 05. mai 2003, 09:45

**Tilstede:** Ole Martin Dahl  
Morten Sporild

**Ikke tilstede:** Trond Christian Frøhaug

**Agenda:** 1. Status  
2. Ukens oppgaver

### **1. Status**

Introduksjonsdelen og inkrement 1 av rapporten er nesten ferdigskrevet. Levert Erik for gjennomsyn.

Fortsatt ikke løst netgruppe problemet..

Trond Christian er fortsatt syk med vannkopper.

### **2. Ukens oppgaver**

Rapportskriving (Ole Martin og Morten)

# STATUSMØTE 1

**Tidspunkt:** 4.februar 2003, 09.00

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild  
Erik Hjemås, veileder

**Agenda:** 1. Status

1. Kravspesifikasjonen blir vanskelig å komme med innen den fristen vi hadde satt oss i følge Gantt skjemaet og forprosjektrapporten. Vi skal prøve å få denne ferdig etter neste utviklerseminar i Stavanger.
2. Erik Hjemås foreslo at vi skulle ta en prat med Tom Røise og diskutere open source som utviklingsmodell, siden denne er litt spesiell og ikke inngår som pensum i faget Systemutvikling 1. Eller studere denne utviklingsmodellen og andre prosjekter som bruker open source. Dette for å få dokumentasjonen vi trenger til prosjektrapporten.
3. Siden skolelinux er blitt et så stort prosjekt, flyter oppgavene litt rundt og "gjørokratiet" hersker. Dvs. den som gjør noe, bestemmer. Dette gjør at oppgaven kan bli endret underveis, og nye oppgaver kan komme til. Eksempel på dette er det ene inkrementet vi fikk utdelt av Skolelinux, automatisk oppretting av mailkonto og oppretting av brukere i LDAP, ble gjort ferdig av Andreas Dahl under helgens samling på Ulsrud. En oppgave vi forøvrig hadde begynt å lese oss opp til.
4. Oppgaven forutsetter at vi lærer oss mye om Linux generellt og om de mange modulene som virker mot hverandre i. Dette medfører i følge Erik Hjemås at vi må dokumentere prosjektet som et forskningsprosjekt i tillegg til et utviklingsprosjekt. Dette har vi også fått beskjed om av Skolelinux, slik at oppgaven kan fortsette etter vi er ferdige med hovedprosjektet.
5. Vi var på utviklerseminar på Ulsrud undomsskole i Oslo i helgen, hvor vi fikk innføring i CVS. Vi opprettet også kontakt med Andreas Dahl (Linpro), som har laget det meste av LDAP- implementasjonen i Skolelinux så langt.

**Gjøvik, 4.februar 2003**

## **STATUSMØTE 3**

**Tidspunkt:** 4.mars 2003, 09.00

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild  
Erik Hjemås, veileder

**Agenda:** 1. Status

1. CUPS støtter ikke LDAP, men dette skal komme i løpet av våren. Vi diskuterte derfor ulike løsninger for styring av skriverkvoter. Erik syntes det beste forslaget var å konfigurere en løsning som allerede eksisterer for styring, PyKota. Denne har ikke ennå kommet i versjon 1.0, men har mye bra funksjonalitet og bruker snmp for kommunikasjon med printer.
2. Vi så gjennom mailinglister for cups, for å finne ut om versjon 1.2 støtter LDAP og når denne er ventet å komme ut. CUPS er planlagt å støtte LDAP, men vi fant ingen spesifikk dokumentasjon på at dette ligger inne ennå.
3. Erik Hjemås skal se gjennom foreløpig dokumentasjon: kravspesifikasjon, statusrapporter og mørerapporter, og komme med tilbakemelding på disse.

**Gjøvik, 4.mars 2003**

## **STATUSMØTE 5**

**Tidspunkt:** 1.april 2003, 09.00

**Tilstede:** Ole Martin Dahl  
Trond Christian Frøhaug  
Morten Sporild  
Erik Hjemås, veileder

**Agenda:** 1. Status

1. Temaer vi må ha med i hovedrapporten:

Om printerproblematikk generellt

Om printerproblematikk i Linux

Hvordan CUPS fungerer (IPP), m/ figur

Hvordan en printerjobb blir sendt gjennom systemet (Skolelinux) med moduler og protokoller.

Figur (bruke dia) foran kapittel om teknologier, med hvordan de ulike teknologiene fungerer opp mot hverandre.

2. Gjøre oss ferdige med testing og konfigurering før påske.

3. Levere foreløpige dokumenter til Erik før påske.

**Gjøvik, 1.april 2003**

## **STATUS MEETING REPORT 18.FEBRUAR 2003**

### **What have we done so far...**

- Developed and published our projects web page at the [www.skolelinux.no](http://www.skolelinux.no) page
- Built the group room
- Installed and configured the test network
- Participated at developer gathering (31.jan-2feb. at Ulsrud VGS.)
- Gathered and read information about PAM, LDAP and CUPS on the Internet

### **The project so far...**

We have read HOWTO's about LDAP, PAM and started to gather information about CUPS. Unfortunately it is not much information available concerning CUPS used together with LDAP. The current release of CUPS, version 1.1.18, don't support LDAP, but it is planned implemented in CUPS v1.2. Version 1.2 is available from CVS on the web. We have downloaded it, but the installation failed.

Concerning the projects goal it has been slightly modified. After talking with our employer it has been clarified that the Norwegian user manual will be written by the student group from NITH. In our pre-project we wrote that this was one of our goals, but it's not anymore.

### **Time limits...**

As a result of the redefinition at the projects technical goal, some changes have been made to the Gantt-chart. The three main-increments in the main section, has been reduced to two.

The main section now consist of the increments: «LDAP structure modification» and «Printing quotas/accounting».

The milestones for «requirement specification» and «Finishing the programming» has been moved forward in time. This is related to the changes in the projects technical goals. For further information confront the Gantt-chart.

### **About the group...**

Because of the weak project description it has been a challenge to define what we are supposed to deliver to our employer. Our project is just a small part of a big open source project, who is probably the main reason why there have been some confusion about who is doing what. We have used the skolelinux mailing list, and this have straighten up some things. But we feel that “the road is being built while we walk” and this can some times lead to frustration for the group members. This has also lead to some demotivation.

### **Contact with the teaching supervisor...**

We have had two meetings with Erik Hjelmås. On this meetings we shortly brief him about current status, what we have done since last meeting and our plans for the next two weeks.

Gjøvik 18.feb, 2003

Trond Christian Frøhaug

## STATUS MEETING REPORT MARCH 18 2003

### What have we done so far since last status meeting...

- Started configuring PyKota for printing quotas.
- Started to find possible net group solutions for LDAP.
- Participated at developer gathering (Feb 21. - Feb 23. at Time high school, Stavanger).
- Started writing about different technologies.
- Installed a network printer in the group room.

### The project so far...

We have decided to try to implement the PyKota quota system. PyKota uses CUPS and makes it possible to store individual printer accounting information for each user. PyKota currently do not have a LDAP backend. It is planned in future versions of PyKota. Jerome Alete, the developer of PyKota, said in a mail that he would be happy to conduct if we decided to make the LDAP backend for PyKota. We have decided that the LDAP backend become a to big task at this time of the project. PyKota is currently using a PostgreSQL database to store quota information for the users. Petter Reinholdsen at Skolelinux said that a PostgreSQL database is need in skolelinux for other tasks also, so we will try to use PostgreSQL as storage backend. Currently we are having problems using the PostgreSQL. We need a way authenticate users in PostgreSQL as the same users that is found in the OpenLDAP directory. This will probably be solved through a PAM module. We have also decided to try to implement netgroups using NSS and LDAP. We have send some mails about this problems and are currently waiting for answers and searching the net for information.

The two originally planed increments has slightly changed. We are now putting all our time into printing and netgroups. We have also written a requirement specification.

### Time limits...

Some changes have been made to the Gantt-chart. We have set the last day of development and testing to April 28<sup>th</sup>. We will try to document the research we have done so far, parallel to developing.

For further information confront the Gantt-chart.

### About the group...

It is a lot of opinions on what we are going to do in our project by people outside our group. It seems that no one really knows what exactly we should do. It much up to us self to decide. We have put our minds to two specific tasks, and will use our time on these tasks. We have realized that research is a big part of our project and we are writing down all the technologies we have read and used. The documentation will become very important for the final report. This because the project is more about research and system administration tasks than code developing.

We are looking forward to many long days of documentation and configuration in the following two months...

### Contact with the teaching supervisor...



We have had four meetings with Erik Hjelmås. On this meetings we have briefed him about current status, what we have done since last meeting and our plans for the next weeks. We have also discussed how our main report should include.

Gjøvik March 18, 2003

Ole Martin Dahl

## STATUS MEETING REPORT APRIL 29th 2003

### What have we done so far since last status meeting...

- Done configuring PyKota for printing quotas.
- Still haven't found any possible netgroup solutions for LDAP.
- TC is ill and might be gone for up to two weeks

### The project so far...

We landed on using PyKota as the solution for printing quota system.

Netgroups using NSS and LDAP still is unsolved. We've had contact with a lot of people through mail ([linuxiskolen@skolelinux.no](mailto:linuxiskolen@skolelinux.no)), and they all have suggestions on why it doesn't work. Dagfinn I. Mannsåker at the University of Oslo, has used the new libnss-ldap module with netgroup support with success but only for one specific machine. We've also googled the net for answers, but only found people with the same problems as us. Libnss-ldap just recently got the LDAP netgroup support, even though rfc 2307 (from march 98) describes it. Therefore we haven't found anyone who has successfully implemented netgroups using LDAP.

The compat module that should be used between nsswitch.conf and LDAP, doesn't support LDAP, NSS/glibc, and need to be re-written for that use. This module might be why we can not seem to get netgroups to work. Since it's only three weeks to project deadline, and according to our Gantt schema we will do no more programming, this module will not be rewritten by us.

We were invited to a gathering at Ulsrud in Oslo this weekend, where skolelinux wanted us to present our finished work for the skolelinux project group. Since we didn't have the time to do this and our deadline isn't until 3 weeks, we agreed to present the project at a later time. The groups from NITH however presented their work since their deadline was before easter.

### Time limits...

Because Trond Christian is sick (chickenpox) and might be that for a while and yesterday was according to the gantt chart the last day for any development, we've decided that both of us should use all our time on writing the report.

### About the group...

We have learned that how netgroups and LDAP work, is not well documented and is quite complex to understand. We would really like to have more time to use on understanding this problem, but time runs out on us.

**Contact with the teaching supervisor...**

We have had six meetings with Erik Hjelmås. On these meetings we have briefed him about current status, what we have done since last meeting and our plans for the next weeks. We have also discussed what our main report should include.

Gjøvik April 29, 2003

Morten Sporild

## Log of hours used in the main project, working as a group.

Day	Date Action	Hours
mo	Project start. Planning the week.	3,0 hours
we	08.01.03 Setting up equipment, installing.	7,0 hours
th	09.01.03 Setting up equipment, installing.	3,0 hours
fr	10.01.03 Setting up equipment, installing.	5,0 hours
mo	13.01.03 Meeting with Knut Yrvin, Frode Jemtland	4,5 hours
we	15.01.03 Pre-engineering report, setting up Pcs from IT-tjenesten	6,5 hours
fr	17.01.03 Pre-engineering report, setting up Pcs from IT-tjenesten	3,5 hours
mo	20.01.03 Pre-engineering report, chosing system engineering model	3,0 hours
tu	21.01.03 Pre-engineering report	7,0 hours
we	22.01.03 Pre-engineering report, meeting with Erik Hjelmås	3,5 hours
th	23.01.03 Pre-engineering report	5,5 hours
fr	24.01.03 Pre-engineering report	4,0 hours
mo	27.01.03 Setting up the system enviroment, (server, thin client server, thin client, workstations, firewall)	3,0 hours
tu	28.01.03 Setting up the system enviroment	7,5 hours
we	29.01.03 Setting up the system enviroment	4,0 hours
th	30.01.03 Setting up the system enviroment	6,0 hours
fr	31.01.03 Planing the weekend, Ulsrud	2.0 hours
sa	01.02.03 Ulsrud	8.0 hours
mo	03.02.03 meeting, report writing	3.0 hours
tu	04.02.03 studying: PAM, openLDAP, NSS, setting up the thin client	7.0 hours
we	05.02.03 studying: PAM, openLDAP, NSS, setting up the thin client	4.5 hours
mo	10.02.03 meeting, planing Stavanger	2.0 hours
tu	11.02.03 Studying skolelinux modules	7.0 hours
we	12.02.03 Studying skolelinux modules	4.5 hours
mo	17.02.03 CUPS, preparing for Stavanger	3.0 hours
tu	18.02.03 CUPS, preparing for Stavanger	7.0 hours
we	19.02.03 CUPS	4.5 hours
fr	20.02.03 Stavanger, leaving at 1300, arriving 1800. Helped set up the Local net	8,0 hours
sa	21.02.03 Coursing etc..	12.0 hours
su	22.02.03 Working on LDAP, arriving Gjøvik 2100.	11.0 hours
mo	24.02.03 meeting, report writing	2,0 hours
tu	25.02.03 CUPS	8,5 hours
we	26.02.03 CUPS	5,0 hours
tu	04.03.03 Status meeting, CUPS, PYKOTA, NETGROUPS	7,0 hours
we	03.03.03 CUPS, PYKOTA, NETGROUPS	4,5 hours
mo	10.03.03 Meeting	2,0 hours
tu	11.03.03 PyKota, Netgroups	7,0 hours
we	12.03.03 PyKota, Netgroups	3,0 hours
mo	17.03.03 PyKota, Netgroups	2,0 hours
tu	18.03.03 Status meeting, PyKota, Netgroups	7,0 hours
we	19.03.03 PyKota, Netgroups	2,0 hours
th	20.03.03 PyKota, Netgroups	3,0 hours
mo	24.03.03 Meeting, PyKota, Netgroups	2,0 hours
tu	25.03.03 Meeting with Erik, PyKota, Netgroups	7.5 hours
we	26.03.03 PyKota, Netgroups	7,0 hours
th	27.03.03 PyKota, Netgroups	3,0 hours
fr	28.03.03 PyKota, Netgroups	4,0 hours
tu	01.04.03 Status meeting, PyKota, Netgroups	7,0 hours

we	02.04.03 PyKota, Netgroups	5,0 hours
th	04.03.03 PyKota, Netgroups	3,0 hours
mo	07.04.03 Monday meeting, PyKota, Netgroups	1,0 hours
we	09.04.03 Netgroups, report writing	7,5 hours
th	10.04.03 Netgroups, report writing	3,0 hours
fr	11.04.03 Netgroups, report writing	5,0 hours
mo	14.04.03 Netgroups, report writing	2,0 hours
tu	15.04.03 Netgroups, report writing	3,0 hours
tu	22.04.03 Meeting, netgroups, report writing	4,0 hours
we	23.04.03 Netgroups, report writing	7,5 hours
th	24.04.03 Netgroups, report writing	4,5 hours

**288.5 hours**

## Log of hours used in the main project, working individually.

Name: Ole Martin Dahl

Day	Date Action	Hours
Mo	06.01.03 Test install of Skolelinux	2,0 hours
We	07.01.03 Pre-engineering report	2,5 hours
Tu	14.01.03 Studying openLDAP	3,0 hours
We	15.01.03 Studying openLDAP	2,5 hours
Mo	20.01.03 Studying openLDAP	1,0 hours
Mo	27.01.03 Studying openLDAP and pdf in linux	5,0 hours
We	29.01.03 Working on webpage	1,0 hours
Fr	31.01.03 Developer seminar at Ulsrud	2,0 hours
Su	02.02.03 Developer seminar at Ulsrud	5,0 hours
Mo	03.02.03 Samba and smbmount	1,0 hours
Th	06.02.03 Studying PAM	5,0 hours
Fr	07.02.03 Studying PAM	2,0 hours
Su	09.02.03 Studing LDAP	1,0 hours
Mo	10.02.03 Studying open source	2,0 hours
Tu	11.02.03 Webpage, Meetingreport and directory services	8,0 hours
Th	20.02.03 Studying CUPS	4,5 hours
Su	23.02.03 Studying Pykota	1,5 hours
Th	27.02.03 CUPS and Pykota testing	4 hours
Mo	03.03.03 Monday meeting and CUPS testing	4,5 hours
Tu	04.03.03 Pykota testing	1 hours
Th	06.03.03 Mail	1 hours
Fr	14.03.03 Report writing	1 hours
We	19.03.03 Status meeting report and PXE studying	2,5 hours
We	26.03.03 Pykota testing	1 hours
Th	27.03.03 Pykota and PostgreSQL	3 hours
Th	03.04.03 Writing about printing	3 hours
Fr	04.04.03 Writing about printing	3 hours
Mo	07.04.03 Thinclient print test	2,0 hours
Tu	08.04.03 Writing about printing	4 hours
Tu	22.04.03 Writing about printing	2,5 hours
Mo	28.04.03 Netgroups	2,5 hours
Tu	29.04.03 Report writing	8,5 hours
We	30.04.03 Report writing	8,5 hours
We	01.05.03 Documenting	2,5 hours
Fr	02.05.03 Documenting	7 hours
Mon	05.05.03 Documenting	7,5 hours
Tue	06.05.03 Testing netgroups	10 hours
We	07.05.03 Documenting	7,5 hours
Th	08.05.03 Documenting	4 hours
Fr	09.05.03 Documenting	7,5 hours
Sa	10.05.03 Documenting	7 hours
Su	11.05.03 Documenting	6,5 hours
Th	15.05.03 Documenting	8 hours
Fr	16.05.03 Documenting	6 hours
		174.5

**Antall timer total: 287.5 + 174.5 = 462**

## Log of hours used in the main project, working individually.

Name: Morten Sporild

Day	Date Action	Hours
Tu	14.04.03 Studying openLDAP	2,0 hours
Th	16.01.03 Studying openLDAP	3,0 hours
Mo	20.01.03 Studying openLDAP	4,0 hours
Th	23.01.03 Studying openLDAP	3,0 hours
Th	30.01.03 Studying openLDAP	3,0 hours
Th	06.02.03 Studying PAM	3,0 hours
Sa	08.02.03 Studying PAM	4,0 hours
Mo	10.02.03 Studying PAM	2,0 hours
Th	14.02.03 Gantt chart	3,0 hours
Fr	15.02.03 CUPS	3,0 hours
Mo	18.02.02 Course material, Time	2,0 hours
We	20.02.03 Course material, Time	3,5 hours
Th	06.03.03 Studying Netgroups	4,0 hours
Mo	10/03/03 Netgroups	2,5 hours
Th	13.03.03 Netgroups	3,0 hours
Sa	15.03.03 Netgroups	2,0 hours
Mo	17.03.03 Netgroups	3,0 hours
Th	20.03.03 Netgroups	3,0 hours
Mo	28.04.03 Meeting, netgroups, report writing	2,0 hours
Tu	29.04.03 Statusmeeting, report writing	8,5 hours
We	30.04.03 Statusmeeting, report writing	6,0 hours
Fr	02.05.03 Report writing	6,0 hours
Su	04.05.03 Report writing	4,0 hours
Mo	05.05.03 Report writing	5,0 hours
Tu	06.05.03 Report writing	10,0 hours
We	07.05.03 Report writing	7,5 hours
Fr	09.05.03 Report writing	7,5 hours
Sa	10.05.03 Report writing	6,0 hours
Su	11.05.03 Report writing	6,5 hours
Tu	13.05.03 Report writing	3,0 hours
Th	15.03.03 Report writing	4,0 hours
fr	16.03.03 Report writing	6,0 hours
		135,0 hours

**Total: 287.5 + 135 = 436.0 hours**

## Log of hours used in the main project, working individually.

Name: Trond Christian Frøhaug

Day	Date Action	Hours
Tu	jan 07 Report reading	2,5 hours
Tu	jan 14 Studyng LDAP	4,5 hours
Th	jan 16 Studyng LDAP	2,0 hours
Mo	jan 20 Studyng LDAP	2,0 hours
We	jan 22 Howto burn Cds in Linux	2,0 hours
Fr	jan 24 Pre-engineering project, backup, smb-mount	1,5 hours
Mo	jan 27 New pr 34 multisession burning, LDAP studying	2,0 hours
Fr	jan 31 Gathering at Ulsrud	2,0 hours
Su	feb 02 Gathering at Ulsrud	5,0 hours
Mo	feb 03 Smb-mount, PAM	2,5 hours
Th	feb 06 Studying PAM	5,0 hours
Mo	feb 10 Meeting report, LDAP+PAM+Gathering	4,0 hours
Tu	feb 11 CUPS, Meeting report. PAM, Mail	8,0 hours
Th	feb 20 Status report ++	4,5 hours
Mo	feb 24 Requirement specification	1,0 hours
Th	feb 27 LDAP, Requirement specification	4,0 hours
Mo	mar 03 Monday meeting ++ (without Morten)	4,0 hours
Tu	mar 04 PostGreSQL + perl-ldap + PyKota	1,5 hours
Th	mar 06 Mail ++	1,0 hours
Mo	mar 10 PostGreSQL + PyKota	1,0 hours
Fr	mar 14 Course in report writing	1,0 hours
We	mar 19 Report writing	2,5 hours
Mo	mar 24 Report writing	1,0 hours
We	mar 26 PyKota config	1,0 hours
Th	mar 27 PyKota + report	3,0 hours
We	apr 02 PyKota conf	1,0 hours
Th	apr 03 Report writing	3,0 hours
Fr	apr 04 Report writing + backup	3,5 hours
Mo	apr 07 LDAP + netgroups	2,0 hours
Tu	apr 08 LDAP + netgroups	4,0 hours
Tu	apr 22 Netgroups	2,0 hours
Th	apr 24 Netgroups	4,0 hours
Th	May 15 Report writing	8,0 hours
fr	May 16 Report writing	6,0 hours
		<hr/> <hr/>
		102 hours

**Total: 102 + 289.5 = 391.5 hours**



## ***Technologies we have used/learned***

### ***Firewall***

A firewall is not a single technology, but a concept. It is an important part of a modern network. The Skolelinux network relies on a router that connects to other networks (i.e. the Internet). The routers task routes all the ip packets that go to or go from the Skolelinux network. It is recommended that this router has a firewall installed. This increase security and make it possible to use the default ip-range settings in Skolelinux.

The Skolelinux network uses private IP-addresses in the range 10.0.2.0-10.0.3.255 and the thin clients from 192.168.0.0-192.168.0.255. The connection to the Internet then relies on NAT (Network Address Translation). NAT makes it possible to use only one public IP-address for all the computers in the Skolelinux network. In our Skolelinux network we have a firewall, that has one NIC that uses the public ip 128.39.80.204. Another NIC on the firewall uses the private IP 10.0.2.1 on the Skolelinux network. 10.0.2.1 then becomes the default gateway for the Skolelinux network. The firewall is then set up to use NAT with iptables[i], a firewall system for Linux.

It isn't only NAT that is important with the firewall, but also the security aspect. It is important to be able to stop unwanted traffic to and from the Skolelinux network. E.g. restricting telnet traffic from and to hosts in the Skolelinux network.

[1] <http://www.netfilter.org/>

## **DHCP**

DHCP (Dynamic Host Configuration Protocol) was created by the Dynamic Host Configuration Working Group of the Internet Engineering Task Force (IETF; a volunteer organization which defines protocols for use on the Internet). [i]

DHCP is a service (protocol) for dynamically addressing IPs to machines on a net. These IPs are taken from a pool of IP-numbers on a server. When a client machine boots or requests an IP in another way, it asks the DHCP-server for a IP-number and gets one from the pool. The identification of this client machine requires a client ID. Most often this is it's MAC (Medium Access Control) which is burned into the network adapter, but DHCP supports other ways of identifying.

The server often also sends this information to the local gateways and DNS servers. A IP is free if it's not in use, and it's more than a certain amount of time since it has been in use. This is called lease-time. Usually the IP given is the same IP it had the last time.

This centralized way of handing out addresses makes it easy to keep notice of which computer has which address, and it's much easier to change the IP range of the net.

In Skolelinux DHCP is used by the thin client server (DHCP daemon, dhcpd) for handing out addresses to the workstations and the thin clients, which needs this when booting directly from the network adapter. This eases Skolelinux philosophy of simply replacing thin clients if there's something wrong with them. The IP range (the IP pool) in Skolelinux goes from 10.0.2.100 to 10.0.3.254 by default.

Ref:

[i] [http://www.dhcp-handbook.com/dhcp\\_faq.html](http://www.dhcp-handbook.com/dhcp_faq.html)

## CVS

CVS is the Concurrent Versions System. This is the most used program for version control systems in open source projects, and can be used for both smaller and larger projects.

- Its client-server access method lets developers access the latest code from anywhere there's an Internet connection.
- Its unreserved check-out model to version control avoids artificial conflicts common with the exclusive check-out model.
- Its client tools are available on most platforms.

[i]

### **More detailed:**

CVS is used to handle all the different versions an open source project makes. Bugs can sneak in, and corrupt the program. With CVS you can go back to see the different versions as it develops, and track errors more easily.

With CVS you can choose to save all versions of the different files. This however is not very economic, since it's a waste of space. Therefore CVS uses a method of saving only the differences between versions.

CVS is great for mutual exclusion. It makes sure that no one overwrites each others changes. First one need to download the working files locally, and when the work is done CVS merges the work of the different developers.

Making CVS started back in 1986 by Dick Grune, with shell scripts. No code from this is in use today, but many of its solutions are. In April, 1989, Brian Berliner designed and coded CVS.

CVS is not a tool for bug tracking. For this Skolelinux uses Bugzilla.

### **Using CVS:**

Download the files :

```
cvs -d :ext:mспорилд@cvs.skolelinux.no:/var/lib/cvs checkout skolelinux/src
```

This adds all the files in /var/lib/cvs to skolelinux/src locally.

To update your own local copy of a file (other developers may

have changed it in the meantime) you use the command update:

```
cv s -d
:ext:msporild@cvs.skolelinux.no:/var/lib/cvs
update Skolelinux/src
```

If you want to see the difference between the files you're working on and the files on the server, use the command diff:

```
cv s -d
:ext:msporild@cvs.skolelinux.no:/var/lib/cvs
diff skolelinux/src
```

To change the files on the server with your local version use

commit:

```
cv s -d
:ext:msporild@cvs.skolelinux.no:/var/lib/cvs
commit skolelinux/src
```

### **Our use of CVS:**

At the first gathering in Ulsrud, we were given passwords and write access to [cvs.skolelinux.no](http://cvs.skolelinux.no). Skolelinux then held a brief course in using CVS, and we spent some time getting to know the system as they claimed we weren't fully members of the project unless we knew how to use this tool.

This access to `cvs.skolelinux.no` has been used to add our website under the `Skolelinux.no` hierarchy, and change information in different files as participants on the gatherings and so on. Later we'll use this write access to add our solutions to the given problems.

The newest versions of some of the modules we needed, PyKota and CUPS, were only available through CVS, so we had to get it there (<http://savannah.nongnu.org/cgi-bin/viewcvs/pykota/>, <http://cvs.easysw.com/cvsweb.cgi/cups/>). This access was anonymous, since we didn't need write access to the files.

Ref:

[i] <http://www.cvshome.org/>

## **PXE**

This is a technology that we learned about at a developer seminar at Time junior high school in Stavanger. This solved the problems with the DHCP connection to the thin clients, in a very smart way.

PXE is an open industry standard, developed by a number of software and hardware vendors. It was initially designed by Intel, with input from several other vendors, including 3Com, HP, Dell, Compaq, and Phoenix Technologies. PXE works with a network interface card (NIC) in the PC, and makes the NIC a boot device. It is trying to be standard technology to make the NIC a boot device. PXE boots the thin client from the network by transferring a “boot image file” from the LTSP server. The LTSP server has a PXE server running. The boot image file is a special thin client Linux kernel that support PXE. Because PXE works with the NIC, it requires a PXE-enabled NIC.

Without PXE the LTSP server needs the MAC addresses of each thin client and the thin client has to have the Linux kernel either on a floppy disk or on a BOOTPROM. The IP address is not enough, because the LTSP server uses DHCP to “distribute” IPs to the thin clients each time they start. PXE makes the configuration and administration of the thin client much easier. Then there is no need for floppy disks or BOOTPROM on the thin clients. Most of the new NIC's support PXE. PXE is available either as a boot ROM chip that you add to the NIC, or as part of the system BIOS if the network interface is on the motherboard. At Time junior high school, it was on a boot ROM chip on the thin clients NICs.

## ***Skolelinux gathering at Ulsrud***

January 31 to February 2 2003 there was a developer gathering at Ulsrud Videregående Skole in Oslo. This was our first meeting with all the other developers in Skolelinux.

### ***Journal***

The first day, January 31:

- Knut Yrvin and Petter Reinholdsen held a brief introduction on what was going to happen this weekend.
- We installed a test network with workstations in three classrooms.

The second day, February 1:

- First there was a course on how to use CVS in the development work of Skolelinux. We got a lecture on how it worked, then each of us tried to add and delete a file in CVS repository. Each of us also got a CVS/SSH account at cvs.skolelinux.no.
- Petter Reinholdsen then held an one hour session about what our group assignment was about. One of the developers, Andreas Dahl, who has and still is working on the LDAP structure in Skolelinux, also came with additions. He drew up how the interaction between the services and how the LDAP was designed. He recommended that we learned about how PAM (Pluggable Authentication Modules) and NSS (Network Service Switch) worked.
- We discussed our assignment with our employer, Frode Jemtland. He recommended that we should use the whole of February to read and learn about the technologies. Then, use March to program and April to test and implement.

The third day, February 2:

- Petter Reinholdsen held a course in how the

Skolelinux CD was built.

- We had a discussion with Frode Jemtland r project. We agreed upon following a strict system engineering model, might be hard in this project. We had to find out a lot about the assignment ourselves, and that the “road is build as we walk”.

### ***Conclusion***

We didn't have much to contribute with at this gathering. The other developers was well into their tasks and had a lot of work to do.

**We did get some information about our project tasks, learned about CVS and met the other developers.**

## ***Skolelinux gathering at Time***

### ***Setting:***

Time junior high school will as one of the first schools in Norway start using Skolelinux. This was about to happen one week after the gathering. Hardware for this has been sponsored by NUUG (Norwegian Unix User Group) and the county council district. The gathering house 30 translators and developers, and 30 teachers and IT-instructors who were at Time for coursing in Skolelinux. The teachers were there to learn about GUI, while the IT-instructors would learn about installation, configuration and general use of Skolelinux.

Our assignments:

*-Friday 21. February*

Set up and install the network in advance of the lectures.

*-Saturday 22. February*

Hold a course for the IT-instructors in installing Skolelinux and set up the thin clients. For the first part we made a installation note.

*-Sunday 23. February*

An assignment that was given to us during the gathering.

We got good feedback on the course we held in the installation of Skolelinux. However the participants also had suggestions on what could be done different, both in the installation and configuration of the thin clients.

### **Experiences from installing Skolelinux at Time Ungdomsskole 22. February:**

\*Good course material (installation).

\*Nice to do things in practice.

\*Committed and engaged mentors.

\*What are the differences between the two language selections during the installation?

\*Bug during installation of ispell (enter must be pressed several times for the opportunity of Norwegian language).



- \*Package error during installation of pr. 36 (task-skolelinux-ltsp).
- \*We had to change the kernel i the configuration file of dhcpd.conf in the ltsp-server, to be able to boot with pxe Ethernet adapter. (PXE didn't work out of the box).
- \*How does one configure the thin clients hardware?
- \*The font daemon of x-windows in some cases didn't run (xfont).
- \*The IT-instructors would like to know more about controlling user permissions, supervision of disk management and who is logged in (who, df, du).
- \*Would have been better to have seen Skolelinux in use, before installing.
- \*There should have been a handed out written document on the thin client installation. This wasn't really our assignment. Since one of the student groups didn't appear at the gathering we agreed to take that part too.

In the Saturday evening, we were given an assignment by Vidar Tyldum (NUUG): try to make a program that would take a semicolon separated file as input, and add these users to the LDAP tree with automatically configuring their profiles, by 3 o'clock Sunday.

More specific:

Find the format of the ldif-schema, which is the file being used to add into the LDAP tree.

Find out if LDAP uses skell.

Make a parser for adding into LDAP.

To make a parser, we had to study the cgi files in the webmin module (the module for graphical system administration tasks in Skolelinux, through a browser).

We made a perl script that created user names from the input file, and figured out where in the different modules we should break in for addition. We would use curl for this (curl is a client to get documents/files from or send documents to a server, using any of the supported protocols (HTTP, HTTPS, FTP, GOPHER, DICT, TELNET, LDAP or FILE). curl offers a bus load of useful tricks like proxy support, user authentication, ftp upload, HTTP post, SSL (https:) connections, cookies, file transfer resume and more).

This was all the time we had, but a main project group from NITH, that we co-operate with, will continue this work.

**Other experiences:**

Few people at the gathering had knowledge or experience with LDAP. Our technical mentor, Andreas Dahl from Linpro, wasn't at the gathering, but we established contact with Jakob Grimstveit who's been working with LDAP in the past and were willing to help if we had questions.

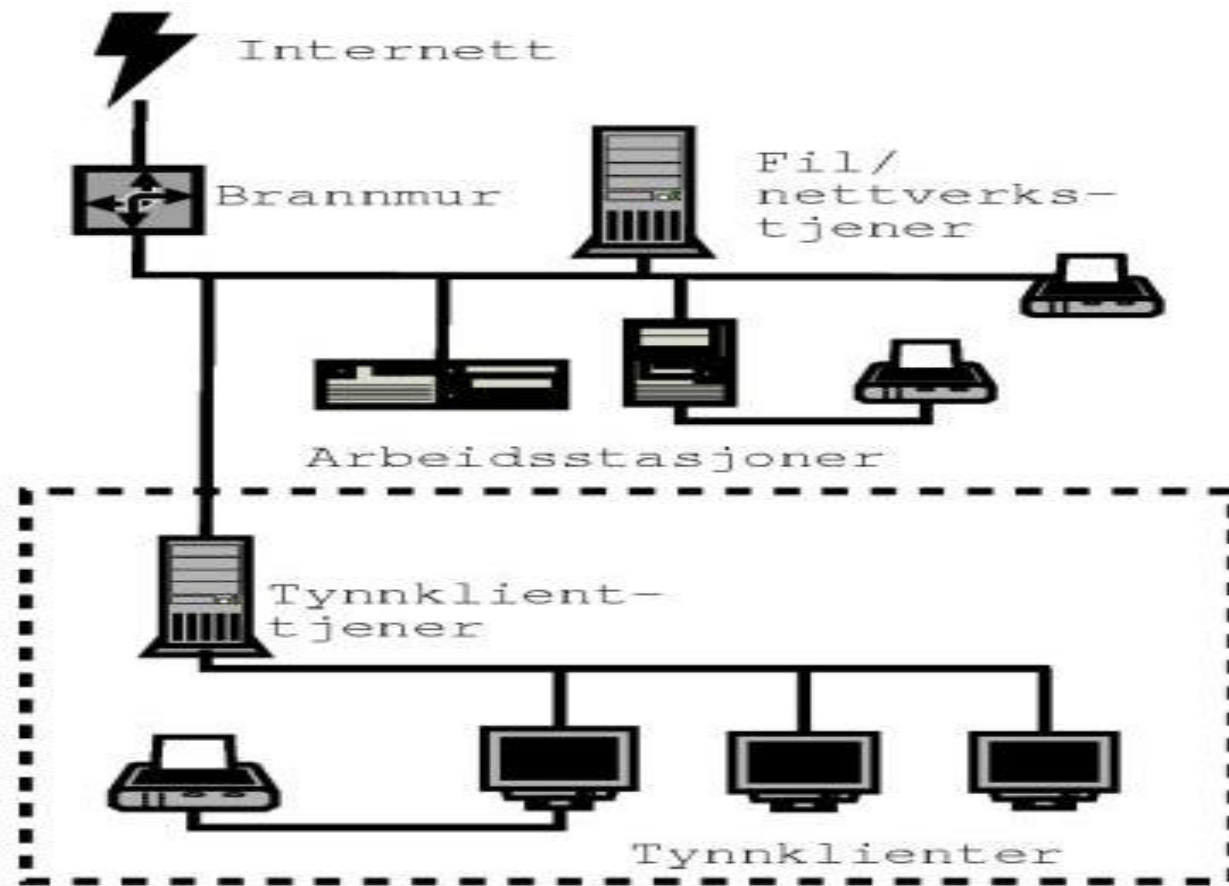
Therefore we didn't exactly get much information directly related to our main project.

“Gjørokratiet” in Skolelinux sometimes makes it confusing to get answers, and to find the right persons. But we got experience in doing a course, and got bugs we could add to Bugzilla.

Appendix:

Installation notes.

# Skolelinux arkitektur



# Skolelinux - installasjon

Legg til CD-Rom som boot i BIOS

Trykk enter for skjermbildet til høyre

Velg språk med space, tab for next.

Velg språk med space, tab for next.

```
Welcome to Skolelinux!

This is a Debian CD-ROM.  Keep it available once you have installed
your system, so you can boot from it to repair the system on your hard
disk if that ever becomes necessary (press <F1> for details).

To start installation of Skolelinux, please press <ENTER>.
Instructions are located in the README.Skolelinux file on this CD-ROM.

If you run into trouble or if you already have questions, press <F1>
for quick installation help.

WARNING: You should completely back up all of your hard disks before
proceeding.  The installation procedure can completely and irreversibly
erase them!  If you haven't made backups yet, remove the CD-ROM
from the drive and press <RESET> or <Control-Alt-Del> to get back to
your old system.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.  For copyright information, press <F10>.

Press <F1> for help, or <ENTER> to boot.

boot: _
```

```
Select locale

< > en_US
< > en_GB
< > en_HK
< > en_IL
< > en_IN

<Next>

Please select which locale you would like to use.
```

```
Select language

< > da (Danish)
< > de (German)
< > es (English)
< > es (Spanish)
< > fr (French)
< > it (Italian)
< > ja (Japanese)
< > ko (Korean)
< > pt_BR (Brazilian Portuguese)
< > ru (Russian)
< > sv (Swedish)

<Next>

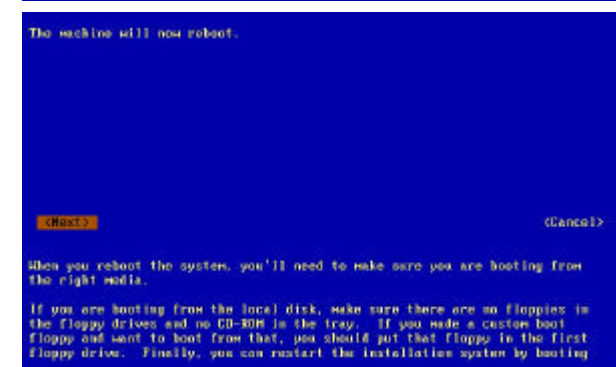
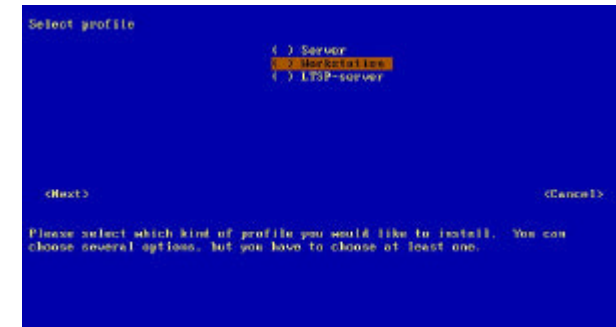
Please select which language you would like to use.
```

# Skolelinux - installasjon

Velg installasjonsprofil

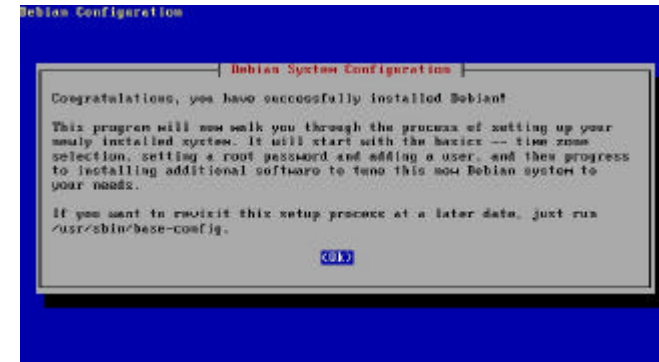
Velg yes, for å partisjonere  
harddisken. Obs! alle data på alle  
partisjoner blir borte.

Start maskinen på nytt.  
Ta ut CD-ROM.

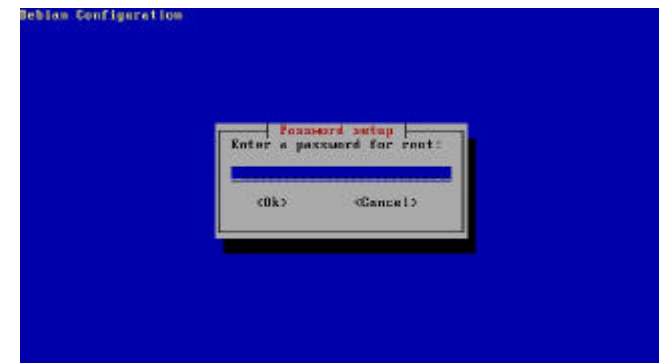


# Skolelinux - installasjon

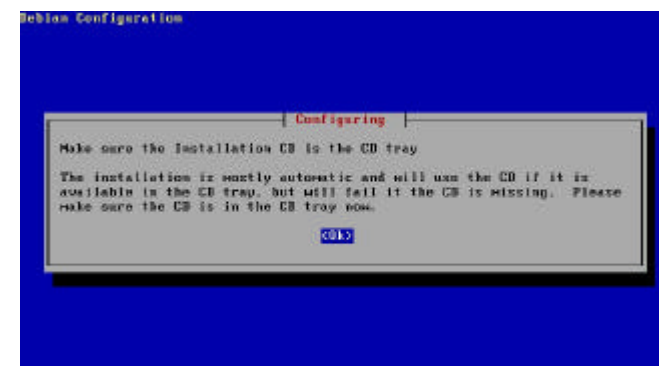
Installasjon ferdig



Sett passord for root (administrator).



Sett inn cd (for konfigurasjon).



Appendix F

# Skolelinux - installasjon

- Konfigurasjon ferdig



## ***Pykota installation guide***

This installation guide is based upon Pykota version 1.00.

To use CUPS on the fileserver we have to configure  
`/etc/cups/cupsd.conf` at the fileserver

```
vi /etc/cups/cupsd.conf
```

Change the "Restrict access to local domain" to  
Allow From All

(This should be set to the host the administrator  
work from not All when the system is in use)

Download PyKota from cvs.

```
cd /tmp
cvs -
d:pserver:anoncvs@subversions.gnu.o
rg:/cvsroot/pykota login
cvs -z3
-d:pserver:anoncvs@subversions.gnu.
org:/cvsroot/pykota co pykota/
```

Install python2.1-dev package.

```
apt-get install python2.1-dev
```

Install postgresql package.

```
apt-get install postgresql
```

Choose LATIN1 as text encoding.

Choose locale as nb\_NO (Norwegian Bokmål)

Install python2.1-egenix-mxdatetime package.

```
apt-get install python2.1-egenix-
mxdatetime
```

Choose yes

Install snmp.

```
apt-get install snmp
```

Choose yes

Install python-pygresql package.

```
apt-get install python-pygresql
```



Install PyKota (tmp/pykota/).

```
python setup.py install
```

Install Pykota PostgreSQL backend

(tmp/pykota/initscripts/). This scripts build the correct database.

```
su postgres
psql template1
\i pykota-postgresql.sql
\q
```

Set the authentication to the database in the

/etc/postgresql/pg\_hba.conf

```
vi /etc/postgresql/pg_hba.conf
set the line to
local          all          trust
and
host           all          127.0.0.1
255.255.255.255 trust
and
host           all          trust
```

Install cupsomatic-ppd package(print driver package)

```
apt-get install cupsomatic-ppd
```

Copy the tmp/pykota/conf/pykota.conf-sample to etc/pykota.conf

```
cp tmp/pykota/conf/pykota.conf-sample etc/cups/pykota.conf
```

Change the pykota.conf to our needs.

```
vi etc/cups/pykota.conf
```

change the administrator name and email

comment the line for printers, uses this file globally for all printers.

change policy for nonexisting users in the database to deny

Add quota script for each printer on the cups server

(etc/cups/ppd).

```
vi etc/cups/ppd/printername.ppd
```

change a line to(near the top of the file):

```
*cupsfilter: "application/vnd.cups-  
postscript 0 /usr/bin/pykota"
```

Now nobody should be able to print on this printer,  
because default deny is set in pykota.conf

Restart the cupsdaemon

```
/etc/cups restart
```

Change the authentication that postgres

uses(etc/postgresql/pg\_hba.conf).

Set the config to accept all users from localhost.

Insert the printer(s) into the PostgreSQL database.

```
edpykota --add -P printername
```

To set qoutas we run edypokata

```
edpykota --add -P printername -S  
softlimit -H hardlimit user1 ...  
userN
```

To see the quota status we run repykota

```
repykota --printer printername
```

or for all printers

```
repykota
```

To make it impossible for ordinary users to set the qoutas we  
change the permissions on warnpykota and edpykota.

warnpykota is the program that sends email to users about the  
quotas.

```
chmod 750 /usr/bin/warnpykota  
/usr/bin/edpykota
```

# PRE-ENGINEERING PROJECT

## SKOLELINUX - USER ADMINISTRATION

January 2003



**By group 8:**

**Morten Sporild**

**Trond Christian Frøhaug**

**Ole Martin Dahl**

# INDEX

<b>1. GOALS AND SETTING</b> .....	3
1.1 BACKGROUND.....	3
1.2 PROJECT GOAL.....	3
1.3 SETTING.....	4
<b>2. EXTENT</b> .....	5
<b>3. PROJECT ORGANIZATION</b> .....	6
3.1 RESPONSIBILITY RELATIONSHIPS.....	6
3.2 OTHER ROLES AND PARTICIPANTS.....	6
<b>4. PLANNING AND REPORTING</b> .....	7
4.1 MAIN CLASSIFICATION OF THE PROJECT.....	7
4.2 DEMANDS OF STATUS MEETINGS AND POINTS OF DECISIONS .....	7
<b>5. ORGANIZATION OF QUALITY ASSURANCE</b> .....	9
5.1 GROUP RULES.....	9
5.2 DOCUMENTS AND OTHER FACTORS.....	9
5.3 BACKUP.....	9
5.4 LOGGING.....	9
<b>6. SCHEDULE</b> .....	10
6.1 LIST OF ACTIVITIES.....	10
6.2 MILESTONES.....	10
6.3 INCREMENTS.....	10
6.4 TIME AND RESOURCE SCHEMES.....	11
<b>REFERENCES</b> .....	12
<b>APPENDIX</b> .....	13

# **1. GOALS AND SETTING**

## **1.1 BACKGROUND**

A “main project” is a final project that every student at Gjøvik College has to carry out in the last term of a three year long Bachelor of Engineering Education. It has a stipulated workload of 18 ECTS points. The project period is from January 2003 with a deadline at May 19 2003.

The official guidelines for the main project says that students are to:

- Execute a larger independent task of interdisciplinary nature.
- Plan, find solutions and produce documentation of these.
- Getting comprehension of advantages and drawbacks of working in a group.
- Realize the importance of making and following-up of plans.
- Getting a positive attitude to method- and problem-oriented way of work and a ability to access different alternatives.

“Skolelinux – user administration” is a part of the bigger Skolelinux project. Skolelinux is a project to create a Linux solution aimed at Norwegian schools. July the 2nd 2001, an initiating meeting was summoned for the Skolelinux project. On this meeting it was decided that Skolelinux should be based on the Debian distribution of Linux. The participants also agreed that the Skolelinux project should provide Norwegian user applications in the two Norwegian dialects, Nynorsk and Bokmål, and in the Sámi language.

Because Skolelinux is a system that will be used by many different users, it also need a module for user administration. This is the main background for our project.

## **1.2 PROJECT GOAL**

### **The project goal**

Make it possible for teachers with minimal computer competence to administrate users in a simple way. Create a simple user manual for the teachers who administrate a Skolelinux network.

### **The technical goal**

Implement a directory service that handles e-mail accounts and printing quotas.

User names, passwords and home area is already implemented in the Skolelinux distribution, but the capability for user groups is missing. The administrators need a way to group accounts in classes, students, teachers, and administrative users.

Another important task is to make the directory service ready for implementation in the Skolelinux distribution.

The technical goal may expand during the project period.

### **The effect goal**

Another goal of the project is also to exploit and develop our competence in both the technical aspect and the aspect of being part of a large system engineering project.

### **1.3 SETTING**

The project is carried out in Gjøvik College (HiG).

HiG has provided us with a group room that we share with one other student group. HiG has also provided us with two computers and each of us have our own computer sited in the group room.

Our teaching supervisor is Erik Hjelmås who is a teacher at HiG. The contact with the Skolelinux organization is maintained by mail, development seminars and meetings.

## **2. EXTENT**

To implement the directory service we will use openLDAP (Lightweight Directory Access Protocol).

Focus of the project is the architecture of LDAP directory service. The GUI is developed by another Skolelinux project group at NITH (Norges Informasjons Teknologiske Høyskole).

The employers demand is that the application must rather be simple and ready to implement, than complex and not fully tested.

### **3. PROJECT ORGANIZATION**

#### **3.1 RESPONSIBILITY RELATIONSHIPS**

Employer: Frode Jemtland, Skolelinux.

Teaching supervisor: Erik Hjelmås, HiG.

The group has chosen a project model without a project leader. This is because of the small size of our group and that we plan to work in a close collaboration. We all take the responsibility for the project becoming a success.

#### **3.2 OTHER ROLES AND PARTICIPANTS**

Group participants: Trond Christian Frøhaug, Ole Martin Dahl and Morten Sporild.

Contact person: Trond Christian Frøhaug

Backup of all the project material: Trond Christian Frøhaug

Updating the web site: Ole Martin Dahl

Keeping the minute book up to date: Morten Sporild



## **4. PLANING AND REPORTING**

### **4.1 MAIN CLASSIFICATION OF THE PROJECT**

The project was divided into four sections:

#### **I Pre-engineering**

- Establish contact with the employer
- Preparation of the group room
- Build “Skolelinux” test network and install the latest release
- Pre-engineering report
- Conferences with our teaching supervisor
- Gather information

#### **II Research**

- Find information and learn about LDAP
- Formulate requirement specification document

#### **III Main section**

- Program the OpenLDAP underlying module structure
- Testing in our own test network
- Report bugs
- Troubleshooting
- Testing in a real “Skolelinux”-environment
- Feedback from teacher supervisor and employer
- Writing Norwegian user manual
- Continuously update the project's website

#### **IV Project closure**

- Finalizing the project report
- Presentation of the project

Because this is a school project we have chosen a incremental system engineering model. Each increment is developed and tested in the main section. We think that this engineering model fits us well because it gives us a well tested, and good product regarding to the employers demand. The model gives us a well documented product, it handles changes and it is controllable. Each increment can also be seen as a finished module.

The system engineering model will bear the stamp of open source development.

The basic idea behind open source is very simple: “when programmers can read, redistribute and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing [1]”. The contact with the other developers in the Skolelinux project is maintained by a mailing list.

## **4.2 DEMANDS OF STATUS MEETINGS AND POINTS OF DECISIONS**

We plan to have one meeting with our teaching supervisor every second Tuesday to discuss status of the project, problems and other challenges.

The contact with other Skolelinux developers will be maintained through a Skolelinux mailing list.

Throughout the project period we have decided to have 7 status meetings with our teaching supervisor. From three of these meetings we will make status reports that will follow a given standard from Gjøvik College.

Status meeting dates:

- February 4, 2003
- February 18, 2003 followed up by a status meeting report.
- March 4, 2003
- March 18, 2003 followed up by a status meeting report.
- April 1, 2003
- April 29, 2003 followed up by a status meeting report.
- May 13, 2003

Major decisions regarding the application will be taken in agreement with our employer. All important decisions should be written down in a log. Meeting reports will be written in turn by each group participants.

## **5. ORGANIZATION OF QUALITY ASSURANCE**

### **5.1 GROUP RULES**

- All expenses must be documented and shared among the group members.
- Each member is bound to meet at times agreed upon. If a group member is ill or can't take part of some other cause, the group member must tell the other members right away.
- Each member is responsible for progress and quality assurance.
- Each group member have professional secrecy regarding personal relationships about the group members.

### **5.2 DOCUMENTS AND OTHER FACTORS**

- All documents has to be written in OpenOffice.org.
- All documents must be stored in the project disk area provided by HiG.
- The pre-engineering project, main project and website will be written in English.
- The user manual (simple edition) will be written in Norwegian.
- All finished documents must be printed out and stored.

### **5.3 BACKUP**

- All work will be stored at:
  - <Not provided by HiG at this time>.
- Backup of our work must be burned out on a CD-R disk each Friday.
- The backup is stored at one of the group members home.

### **5.4 LOGGING**

- All configuration and programming must be commented and logged.
- All meeting reports must be logged.

## 6. SCHEDULE

### 6.1 LIST OF ACTIVITIES

- Preparation of the group room.
- Develop and maintain website.
- Installation of software.
- Finishing pre-engineering project.
- Develop requirement specification.
- Research
- Program the OpenLDAP underlying module structure.
  - LDAP structure modification.
  - Email accounts.
  - Printing quotas
- Report writing
- Testing and bug fixing.
- Develop the Norwegian user manual.
- Finish the main project report.
- Prepare the presentation.
- Present the project.

### 6.2 MILESTONES

- Hand-in the pre-engineering project. **January 24, 2003.**
- Finish the requirement specification. **February 4, 2003.**
- Finish the programming. **April 2, 2003.**
- Finish the LDAP module, Norwegian user manual. **May 12, 2003.**
- Complete the main project report. **May 16, 2003.**
- Present the project. **May 26, 2003.**

### 6.3 INCREMENTS

The development section will be divided into 3 increments.

- **Increment 1 LDAP structure modification.**

- Group accounts in classes, students, teachers, and administrative users.
- Test the increment.
- Document the implementation.
- **Increment 2 Implement e-mail accounts.**
  - Implement e-mail accounts for the different users.
  - Test the increment.
  - Document the implementation.
- **Increment 3 Implement printing quotas.**
  - Implement printing quotas for the different users
  - Test the increment.
  - Document the implementation.

## **6.4 TIME AND RESOURCE SCHEMES**

All the time limits from the time scheme (Gantt-chart) should be observed as far as possible. The most important is to keep the time limits regarding to delivering the hand-ins. The Gantt-chart will be updated during the project period. See Gantt-chart.

## ***REFERENCES***

[1] [www.opensource.org](http://www.opensource.org)

## PROSJEKTAVTALE

mellom Høgskolen i Gjøvik og oppdragsgiver Linux i Skolen

for hovedprosjektet Skolelinux – User administration

utført av studentene Trond Christian Frøhaug, Ole Martin Dahl, Morten Sporild

Avtalepartenes plikter vedrørende gjennomføringen av prosjektet og rettigheter til anvendelse av de resultater som prosjektet frembringer;

1. Studenten(e) skal gjennomføre prosjektet i perioden fra 20030101 til 20030518  
Studentene skal i denne perioden følge en oppsatt fremdriftsplan der HiG yter veiledning.  
Oppdragsgiver yter avtalt prosjektbistand til fastsatte tider.  
Oppdragsgiver stiller til rådighet kunnskap og materiale som er nødvendig for å få gjennomført prosjektet.  
Det forutsettes at de gitte problemstillinger det arbeides med er aktuelle og på et nivå tilpasset studentenes faglige kunnskaper.  
Oppdragsgiver plikter på forespørsel fra HiG å stille med sensor vederlagsfritt. Vedkommende person må tilfredsstille kravene i "Forskrift om eksamen ved Høgskolen i Gjøvik" §10.4 (ekstern sensor).
2. Kostnadene ved gjennomføringen av prosjektet dekkes på følgende måte:
  - Oppdragsgiver dekker selv gjennomføringen av prosjektet når det gjelder f.eks. materiell, telefon/fax, reiser, nødvendig overnatting på steder langt fra HiG. Studentene dekker utgifter for trykking og ferdigstilling av den skriftlige besvarelsen vedrørende prosjektet.
  - Eiendomsretten til eventuell prototyp tilfaller den som har betalt de komponenter og det materiell osv. som er brukt til prototypen. Dersom der er nødvendig med større og/eller spesielle investeringer for å få gjennomført prosjektet, må det gjøres en egen avtale mellom partene om eventuell kostnadsfordeling og eiendomsrett.
3. HiG står ikke som garantist for at det oppdragsgiver har bestilt fungerer etter hensikten, ei heller at prosjektet blir fullført. Prosjektet må anses som en eksamensrelatert oppgave som blir bedømt av faglærer/veileder og sensor. Likevel er det en forpliktelse for utøverne av prosjektet å fullføre dette til avtalte spesifikasjoner, funksjonsnivå og tider.
4. Den totale besvarelsen med tegninger, modeller og apparatur så vel som programlisting, kildekode, disketter, taper etc. som inngår som del av eller vedlegg til besvarelsen, blir HiG's eiendom og kan vederlagsfritt benyttes av HiG til undervisnings- og forskningsformål. Besvarelsen, eller vedlegg til den, må ikke nyttes av HiG til andre formål, og ikke overlates til utenforstående uten etter avtale med de øvrige parter i denne avtalen. Dette gjelder også firmaer hvor ansatte ved HiG og/eller studenter har interesser.  
Besvarelser med karakter 2.5 eller bedre registreres og plasseres i skolens bibliotek. Det legges også ut en elektronisk prosjektbesvarelse uten vedlegg på skolens Internett-side under biblioteket. Studentene skriver under på en kontrakt hvor de gir tillatelse til at deres hovedprosjekt blir publisert i papir og nettutgave (jfr. Lov om opphavsrett).

5. Besvarelsens spesifikasjoner og resultat er oppdragsgivers eiendom, og kan anvendes i oppdragsgivers egen virksomhet. Gjør studenten i sin besvarelse, eller under arbeidet med den, en patentbar oppfinnelse, gjelder i forholdet mellom oppdragsgiver og student bestemmelsene i Lov om retten til oppfinnelser av 17. april 1970, §§ 4-10.
6. Studenten har ikke rett til å publisere sin besvarelse, det være seg helt eller delvis eller som del i annet arbeide, uten samtykke fra oppdragsgiver. Tilsvarende samtykke må foreligge i forholdet mellom student og faglærer/veileder for det materiale faglærer/veileder stiller til disposisjon.
7. Studenten leverer 3 - tre - eksemplarer av oppgavebesvarelsen med vedlegg til bedømmelse, hvorav oppdragsgiver skal ha en. HiG kan stille til disposisjon ytterligere eksemplar(er) for oppdragsgiver mot at denne godtgjør produksjonskostnader.
8. Denne avtalen utferdiges med et eksemplar til hver av partene. På vegne av HiG er det avdelingens kontorsjef som godkjenner avtalen.
9. I det enkelte tilfelle kan det inngås egen avtale mellom oppdragsgiver, student(er) og HiG som nærmere regulerer forhold vedrørende bl.a. konfidensialitet, kostnadsdekning og økonomisk utnyttelse av resultatene.  
  
Dersom oppdragsgiver og student(er) ønsker en videre eller ny avtale, skjer dette uten HiG som partner.
10. Eventuell uenighet vedrørende forståelse av denne avtale løses ved forhandlinger avtalepartene i mellom. Dersom det ikke oppnås enighet, er partene enige om at tvisten løses av voldgift, etter bestemmelsene i tvistemålsloven av 13.8.1915 nr. 6, kap 32.

Studentene: Olav Kjetil Skjold  
Jens Ole Engelsen  
Morten Espenild

Gjøvik den 30.01.2002

Faglærer / veileder: Leivt Skjelvåg

Torstein Skarvane  
Oppdragsgiver

[Signature]  
Kontorsjef