University of Bremen, Informatics Department

Bachelor-Report
User management in the LSOC system

Project
Location Sensitive Office Communication

Author:
Georgi Parvanov
Informatics
Matr.Nr. 1889901

Bremen
16.06.2008

Ich erkläre hiermit, dass ich diese Arbeit selbständig durchgeführt habe und keine außer den angegebenen Hilfsmitteln und Quellen verwendet habe.

Datum: _____ Georgi Parvanov _____

Abstract

As part of the LSOC-Project (Location Sensitive Office Communication Project) of the University of Bremen, the report develops the user management and user access rights of the project.

Beginning with a brief description of the project and its structural model, the report goes on to present the main concept and aim of the project as well as to explain the followed general approach. It then describes the hardware and software used in the project as well as their configuration. Since the management of personal data is an inseparable part of user management, the report also addresses the vital question of data privacy and gives an account of the main data privacy regulations of the University of Bremen, where the project will be implemented. It further describes in detail the user management model of the project, which divides users into groups according to their status in the system and defines their access rights. A common method of identification and authorization is also described and developed.

The report introduces a user management system divided into two modules: one managing the creation and editing of user profiles and another managing the groups and their members.

1. Introduction

The LSOC Project is a part of a smart building concept based on numerous touch screens. The team of LSOC consists of 15 Students. Professor Christian Freksa, Ph. D., Kai-Florian Richter and Falko Schmid support them.

The main idea is to develop a network of touch screens with small sized personal computers and a server for storing data. The implemented software should be able to manage the network and supply the user with certain functionality. This report presents the user management of the project. The system has several groups of users such as administrators, users and guests. Some users, who are employees and work in the building, have more privileges in the system than others. The project is plugin oriented for more flexibility and so is the management. It is divided in two plugins, one for managing the users' data and one for managing the groups of users. The data storing in the project is realized by XML configuration files, a MySQL Database and OpenLDAP. The user management uses the MySQL for users' non sensitive information and OpenLDAP for sensitive information.

In this chapter is described the concept of the project, the architecture and the hardware and software used for the creation of the system. At the end the design and graphical user interface are shown.

1.1. About LSOC

LSOC (LOCATION SENSITIVE OFFICE COMMUNICATION) is a students project at the University of Bremen under the direction of Prof. Dr. Christian Freksa. The project started October 2006 and will be finished in summer 2008.

1.1.1. LSOC Concept

The project has the goal to implement a location based communication possibility in the new building of the computer science department of the University of Bremen. This platform is based on a number of touch screen devices and provides a possibility for users to communicate and publish information within the building. This goal is accomplished by building a java-based control program running on several touch screen panels, which are going to be distributed evenly inside the building.

1.1.2. LSOC model

At these touch screen panels the employees as well as visitors will be able to control every function the building has to offer (controlling lights, windows, roller blinds), depending on each users' status.

Its real benefit lies in another feature, though: users are able to write their own programs that can be combined with our system. These programs are called plugins (see. [LSOCPR]).

These plugins include:

- Video-Conferencing between panels and also between a panel and an employees' computer running our system
- A virtual blackboard, where people can post messages for everyone or send a short message to a different panel. You can upload videos taken with the built-in web camera for everyone to see immediately, too.
- An electronic map, which shows the whole building. You can browse the different levels, zoom in and out. You can also perform a search to look up an employees' office location quickly and without effort
- A simple interface to control every single light and window inside the building depending on the status a given person has (user, administrator, employee, etc.)
- A java-based browser, where you can surf the internet and save files to the panel's built-in hard disk drive

1.1.3. LSOC architecture

Program structure

The Program structure is designed simple, but is meant to ensure as much prospects for extension as possible. The best way, following the innovation in the software engineering, is to do a system based on plugins.

The system has a core, which is initialized when the system is started. The core itself initializes the Graphical User Interface and starts multiple services such as a cron, whose responsibilities include starting and stopping plugins. All the functions of the system are realized with plugins. That way the extension abilities and the flexibility of the system are assured.

1.2. Hardware

1.2.1. Hardware Specifications

A custom small sized PC Mini ITX with 12.1" TFT-LCD touch screen module. For more information see Addendum (Chapter 1. Hardware).

1.2.2. Placing the units and electricity

The touch screens are intended evenly distributed, but also where needed. There will be a panel at every office and two big size touch screens in the lounge to welcome visitors of the Cartesium and to offer services such as searching an office. The big screens are intended for the use by visitors of the Cartesium.

1.3. Software

1.3.1. Operating system – Linux

There are many reasons which have provoked and stimulated our choice of Linux for the project. To begin with is the fact, that Linux is small and needs little resources. On the other hand it is powerful enough for the needs of the project. And of course the fact that it is available for free cannot be underestimated, considering that the project is on a budget from the University of Bremen. As a free operating

system Linux has different versions, from which we had to chose the one with the best qualities for the needs of the project. We choose Xubuntu version 7.10 Desktop i386 with Java 6 environment, because it fulfils all our needs and it is fully compatible with the hardware.

1.3.2. JAVA

Next to a good and suitable operating system, good portability was the second thing that we needed most for the project. The portability plays the main part for the projects ability of being used in different environments. It is also needed if the software should be executable on different types of computers with different operating systems as it is intended to be the case.

These are the reasons which made JAVA and the Java Virtual Machine the first and clear choice for our project. In the last few years JAVA has proven itself for being very portable as it is been used successfully in an enormous variety of devices such as music players, mobile phones, Internet and personal computers.

1.3.3. OpenLDAP and MySQL

MySQL and OpenLDAP are used for data storing. For the user management the more relevant part is the OpenLDAP, because it contains all sensitive information. “OpenLDAP Software is an open source implementation of the Lightweight Directory Access Protocol”. LDAP is based on a client-server model and is intended to work more in read than in write modus. OpenLDAP Software is derived from University of Michigan LDAP 3.3 Release (see. [OLDAP]). It is based on the X.500 Directory Access Protocol (DAP) and it can be easily migrated to other Directory Services for storing the user data. The OpenLDAP is community developed LDAP software and it is free to use. There are two major suppliers of OpenLDAP libraries for use with Java. One is from Netscape: Netscape LDAP SDK (see. [NTSCA]) and the other, officially recommended by the OpenLDAP community, is JLDAP developed by Novell (see. [NOVELL]). For our purposes the recommended one, from Novell, was chosen.

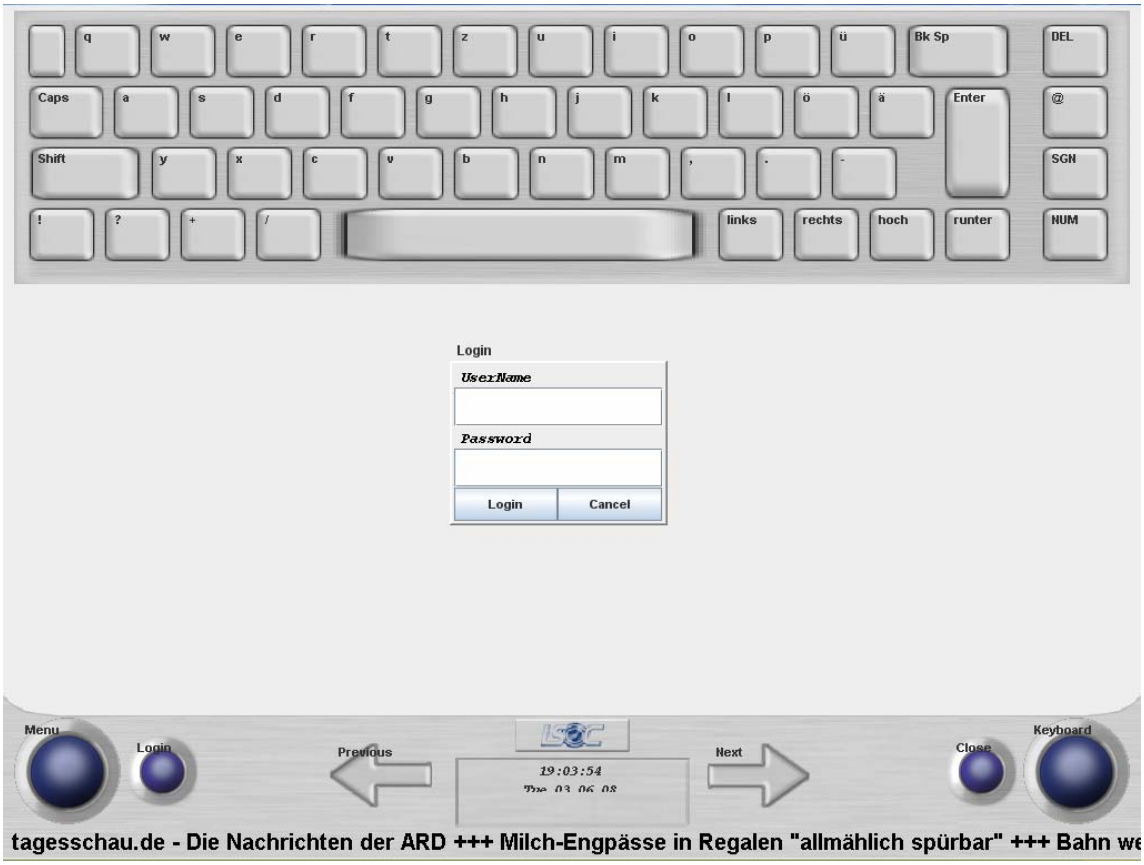
OpenLDAP is an implementation of a directory access protocol and this means that the structure of the data storage is directory oriented. A directory is like a database, but contains more descriptive, attribute-based information. A directory structure can be approached as a tree. It has one root (a starting point) and numerous subdirectories. This directory model is based on *entries* that have a *Distinguished Name* (DN) and set of attributes. Each attribute has a type and one or more values. In these attributes it is possible to save any kind of information in binary format. For example it can be a text (e-mail) or photo (written as binary data). The support of binary information gives the freedom to save any kind of data (see. [OLDAP]).

For the OpenLDAP Server the standard Oracle Berkeley database is used. Also known as BDB it is an open source, embeddable database with very good performance, reliability, scalability, and availability for application use cases that do not require SQL. It ensures a fast and reliable, local persistence with zero administration database. Zero administration means that after installing and configuring the database, no more administration will be needed and the database will work without any maintenance. [ORACLE].

The second database is MySQL. In the project many settings and configuration data are saved in SQL Tables. User insensitive data information is also stored in those tables for more flexibility. The users' information is divided in two different servers, so if there is break in one of the servers it would not be possible to retain all the user information. MySQL is used for the SQL and the reason for that choice can be directly quoted from a statement of Sun Microsystems itself: "The world's most popular open source database."[MySQL]

1.3.4. Design and graphical user interface

The design is intended to be user friendly and comfortable as possible. Besides all other means also it needs to be usable by performing operations on touch screens. Furthermore there is one window focus multitasking environment. That means, there can be worked with more than one Application, also known as plugin, but only one of them has focus, which connotes, that only one is visible to the user at a given moment of time (see Figure 1). Also in the sense of providing comfort to the users the interface of the system is scalable, which means that touch screens with different size and resolution can be used. The system is optimized to work with a standard screen resolution of 800x600.



(Figure 1. LSOC Interface, Login Module)

2. The User Management in LSOC

The chapter treats the data privacy in general and in relation with the project. Furthermore it describes the user management platform and the principals onto which it has been build. Next there comes the introduction and description of the roles, their properties and credentials. To manage the users, who are separated in groups, two plugins and a LDAP handler are implemented, which have the task to complete the data transfer between the servers and the panels (see 2.3.3. LDAP Handler). At the end of the chapter the process of user authorization is described.

2.1. Data privacy

The LDAP server is the mean, which provides protection of the user's personal data. Setting up a LDAP server and by this means providing a good, reliable and secure user management is of great importance for the LSOC project, considering the fact that the system contains personal data, such as name, surname, user account, password etc., for every user.

Knowing the power information holds upon every member of modern society it is inevitable having to straighten up the very concept of "data security" and the principles of data protection which University of Bremen, respectively our LSOC Project obeys.

According to the German data protection legislation, data protection means "to ensure against the misuse of personal data during storage, communication, modification and erasure (data processing) and thereby prevent harm to any personal interest that warrants protection". [Flaherty'92]

Every land has its own laws, issued to secure that personal data is only accessible to those whom it concern and, thus, gives individuals the certainty that inaccuracies would be either prevented, compensated or punished. Data protection is not only of great importance for credit checking purposes, but also for each organization or project using or based on the usage of personal data, for it allows fair treatment of the individuals. It permits only legitimate and lawful reasons for processing personal information and forbids its sharing [Bennett'92].

There are several standards, which regulate the question of data security on international levels such as "The International Standard ISO/IEC 27002 (17799)" [ISO/IEC]. This is a comprehensive Information Security Management Standard, which covers the topics data security and information protection and sets very clear rules about the way private information is supposed to be handled.

Like every other country, Germany has its own standards about data security, which is coordinated with the international norms concerning that delicate matter, and all institutions and organizations have their own standards, which obey to those of German and international level.

The University of Bremen has strict rules about data security, which concern equally students and professors respectively, tutors, derived from the "Bremisches Datenschutzgesetz" and "Bremisches Hochschulgesetz"). [DataSec]

Regarding the matter of data security at the university of Bremen, there surely has to be mentioned its "Kursmanagementsystem" Stud.IP.

This management system is created to facilitate the work of students and professors, allowing them personal access to all the information and materials, which are to be used for learning purposes. To get an account to the system every user (professor, tutor or a student) has to give his name, surname and email

address. This data is allowed to be used only by the legitimized persons from the University of Bremen and not to be given away to a third person. So the university has a good and effective strategy of protecting personal data and the LSOC Project follows this good example.

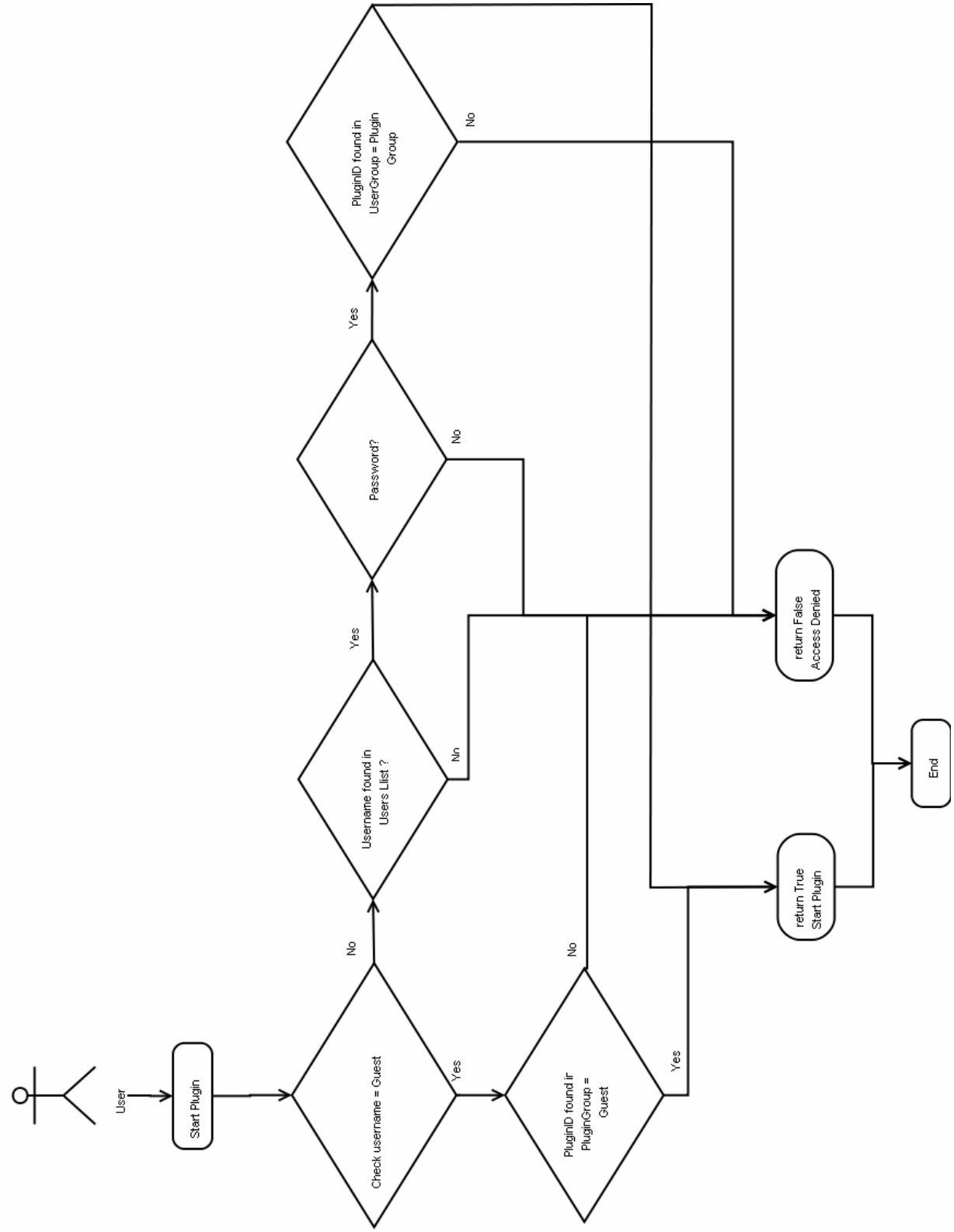
2.2. User management

The user management of the LSOC system is meant to be working on two databases. On one hand there is the LDAP server, needed and created for holding the users' sensitive information. On the other hand we have a MySQL database that is where all the other information (which has no need of privacy protection) is stored. In general there are three roles on the LSOC system (see: 2.2.1 Roles) and, therefore, we have three basic user groups. Other than these three basic groups it is possible to create many custom groups as it is it is possible, that each registered user creates as many and as different groups as wanted. The users can create groups to combine other users into working groups that can have more or less credentials for running specific plugins on the system, specified by an administrator. By creating custom groups different levels of rights can be archived. And it would be possible for the members of the group to use a custom plugin that is not allowed for all users. The plugins are also separated in groups, so that the users from each specific group can run only the plugins from their plugins group.

Supposed that only one person can use the touch screen at a time, there can only be one user who is logged on the system at that particular moment of time, thus, there is also just one variable in the system, which holds the username of the logged user at the particular moment. The parameters, which are received by the Login method, are username and password. The method checks whether both parameters match with the profile in the LDAP server and if they do, the variable LoggedUser is value gets set to the value of the username. If no match can be found between both parameters, which is in the cases "username not found" or "wrong password", the method returns false and the variable does not get changed.

The Logout method, on the other side, can only change the value of the LoggedUser to the value of a Guest. By changing this value the Logout method practically can only get the user logged out of the system.

By starting the system the variable LoggedUser gets set to Guest. The core of the system calls the Authorize Method to authorize the run of a plugin, it proves if a certain plugin may be run or not. The Method checks the variable LoggedUser and if its value is Guest it searches for the plugin ID in the Guest Group of the plugin groups. If the ID is found in the plugins Guest groups at the LDAP server, an answer "true" is returned and the core will be allowed to run the plugin. If the plugin does not get found in the group, "false" will be returned and the core will not run the plugin (see Figure 4. Authorize Method).



(Figure 4. Authorize Method, package: levelo.core*)*

For achieving flexibility and allowing expandability the user rights are controlled by allowing or denying access to execute different plugins. The administrators of the system are those who make the decision if a certain plugin can be used by anyone or whether is it a plugin with special purpose, which can be used only by a specific group of users. Based on these criteria the plugins are sorted in the appropriate group.

The system is module oriented and allows starting plugins only from the main plugin Menu, where those plugins, which are not allowed, are also not shown. Thus they cannot be seen by the user and the user rights are accomplished also in this module manner.

As the plugin group of the administrators consists of all available plugins, when logged in as administrator, all the plugins are visible in the plugin Menu. Since only the administrators are allowed to create new users, the User Manager plugin is only shown in the administrators menu. The other plugin for administration purposes is the Group Manager plugin and unlike the User Manager plugin, this one is visible not only to the administrators, but it is also allowed for registered users. Registered users however may use the Group Manager with some restrictions. The registered and logged users can run it, but they can only see or change custom groups, owned by them. As mentioned above they can create a group and add or remove members, but they neither can see, nor can they change the contents of the three standard groups, which are “administrators”, “users” and “guests” (see Table 1. User Rights).

The Guest group is technically called a group, because it consists only of one user, but in spite of this fact it has not only one plugin, but a whole group of plugins that can be changed by the administrators.

Further more there are some internal functions of the LDAP Handler, which are to be used by the User and the Group Manager plugins and are also accessible for the other plugins in the system (Addendum Chapter 2.1 API Internal Interfaces). These methods include methods for reading, writing and deleting a user in the OpenLDAP Server. The methods that read and return general information are for example those that prove if a user name is available on the server or whether it is already taken. They also read the group of a particular user and the group of a particular plugin. There are furthermore Methods, which read the lists of all plugins installed, all users, members of groups and others. Those return only names with no sensitive information and are used by the management plugins for example the User Manager and the Group Manager plugins (see Addendum, 2.1 API Internal Interface). There are also Methods, used to arrange the plugins in the standard and the created custom groups etc. (For more information see the Addendum)

The plugins are responsible for the proper usage of user information. For example, if a user is logged on to a panel that he does not own, the plugins on their side are only allowed to change this information on the Server, which can later be used by the owned Panel (for example the Calendar plugin). If user A is logged on to a panel owned by user B and tries to change Calendar entries, the plugin has to ensure that the Calendar owned by user A will be changed, and not the one owned by user B. This way, more mobility for the users is achieved, when they want or have to work on the way.

2.2.1. LDAP Handler

The LDAP Handler is one of the most important implementation parts of the user management. It includes the basic methods, which are used by the plugins to access the LDAP server. The low level operations are done by the Novell JLDAP library. This includes building the connection to the server, reading or writing changes and closing the connection. The LDAP handler itself carries out the interface between the plugins and the low level library. It has different methods for reading and writing user information in the LDAP server. It also accomplishes the connection between the user data in the LDAP server and the MySQL server with the help of the User ID property. It reads the user data from both servers and combines it to a user object, which is later used by the plugins. For example, the User Manager plugin uses the LDAP handler to load and save all user data, whenever a user is created, deleted or searched for. The Group Manager plugin uses the handler to load, change and save all the lists of the users and groups (for more information see Addendum Chapter 2.1 API Internal Interfaces).

2.2.2. Login and Logout

The Login Procedure is an action performed by the user with the purpose of changing his status in the system, giving him an access to his account. This procedure changes one is status in the system from that of an ordinary “guest” to that of a logged-in user, thus supplying him with certain rights, which depend on his credentials. For example, when a user is logged in to the system, he is allowed to start plugins from his group. Or he has the right of using the Group manager to create his own user groups and add or remove members from them.

In contrast to the login procedure, the Logout procedure changes the system status of a person from a logged in user to that of a guest. As they have no accounts guests do not possess any credentials, so they can only use the plugins intended for free use.

2.2.3. Roles

By developing a user management system we are using the term “Role” for describing the general types of users that are possible to exist in the system. In roles are combined all users with the same rights.

- Administrator

The role of the administrator is to provide the maintenance of the system. The user management is designed to allow for having more than one administrator. If needed, members can be added to the group of administrators. Only one account can not be deleted and that is the one of the administrator of the LDAP Server. If his account would be deleted the system would get unreachable for everyone. As the system grows and gets more members in a later phase, more administrators might be needed to examine the plugins and maintain the users groups. The members of the administrator group can run and use all available plugins on the system with no restriction whatsoever.

- User

The role of the User is the general case of a user of the system. This role is planned as a starting point for all the personnel working in the Cartesium and using the system. The users can run and use the plugins allowed by the administrator; only

the administrator can allow changing the plugins groups for security reasons. Although users can create custom groups and add members to the group, they do not have the rights to change the plugins group. They must contact an administrator if they want to change the allowed plugins (remove old or install new plugins).

- Guest

The role of the guest account is rather simple. It is intended to be used when no registered user is logged in. The system will always start with the guest account. On the LDAP Server a plugin group named Guest is created. The administrator can decide which plugins may be put in this group. In general, these will be plugins, such as Map, for searching an office or a person, news, blackboard, browser etc.

2.2.4. User Object Properties

Here is the description the user properties, used by the project. Those properties can be extended if needed. Both databases are flexible and as far as the connection user ID is preserved they can be extended without any limitations what so ever.

- UserID
  The User ID is an identifying number. It is generated in the MySQL Table and is unique. It is also written and saved in the LDAP Server. With the help of this unique number the User information from the MySQL server and the LDAP server is combined in one User object, which contains all existing information about the user.
- Login
  Login or also known as username. In the LDAP Server it is represented as a uid attribute.
- Password
  The password of the user is meant to identify his/her personality against the system.
- Firstname
  That is the first name of the user.
- Surname
  Last name or family name of the user.
- Title
  The title of the user such as Mr., Ms., Prof., Dr., etc.
- userpanels_id
  The ID numbers of the Panels owned by the user.
- accounttype
  It represents the account type. It represents the user group (for example administrator, user, guest).
- workgroup_id
  That is the workgroup of the user, which in the general case is the same group as in the LDAP user group.
- address_id;
  It represents the office number.
- email;
  Business E-mail of the user

2.2.5. Rights

The user rights represent a certain number of allowed actions in the system, which are controlled by the administrators by setting certain number of restrictions in the profile of each user. Those restrictions concern the possibility of creating, modifying and/or deleting a user on one side and start particular plugins on the other. In Table 1 (User Rights) the rights are sorted with respect to the user role in the system. With an "X" are marked the allowed actions for the user. As a summary of the table it can be said, that the administrators are allowed to execute every possible action in the system. The user can create and manage custom groups and the guests can only use public plugins.

Rights\ Role:	Admin	User	Guest
Create/Edit/Delete a User Account	X		
Create, Modify or Delete Any Group	X		
Create Groups/Assign Users/Remove Users	X	X	
Users Can Modify Their Own Profiles	X	X	
View public User and Group Information	X	X	X

(Table 1. User Rights)

2.2.6. Use cases

The use cases are created to explain the user management execution and flow. The following introduction illustrates some of the possible situations for the performance of the program.

There is the case when a guest wants to run a browser plugin. He gets the rights from the program and can click on the browser button in the menu. Then, after a credential check, the browser plugin can be run. But if a guest wants to run a group manager plugin, which he is not be allowed to do, this plugin is not shown in the Menu with Guest login and it can not be started.

In case that a certain User A, who is a logged in user and an owner of the LSOC group for example, wants to run a plugin from the Guest plugin Group (like a browser or a map), he is allowed to do it and the plugin will be started. For this action the program gives an answer "true", because for running a plugin from the Guest Group, no credential is needed. Also if this User A wants to run a Group Manager plugin, he will again be given permission. This time the credentials of the user will be checked, the Group Manager will be started, but only those groups owned by User A will be shown with the option to create a new group.

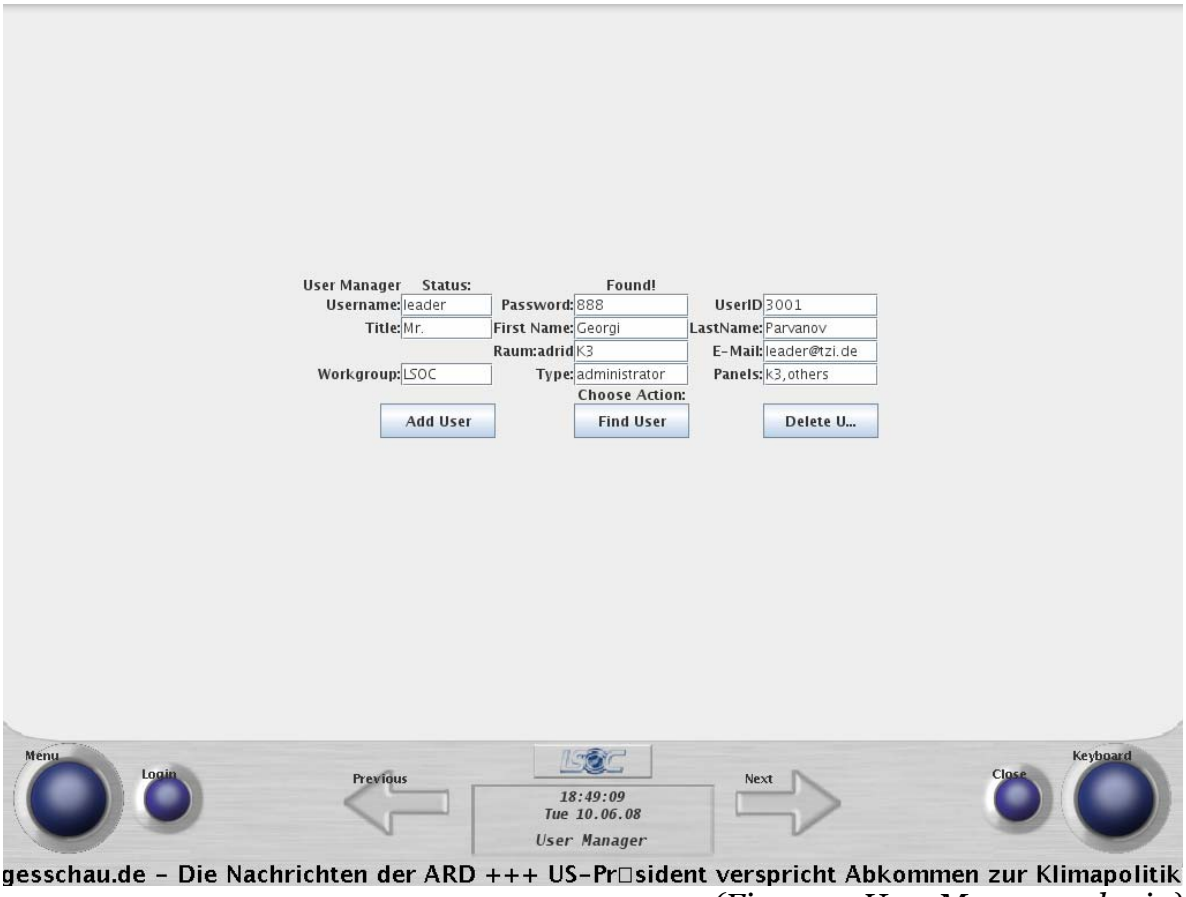
The administrator and all users in the Administrators Group have privileged credentials, so that the cases connected with them would have only one possible answer from the program and that is "true". Like for example in the situation when an "admin" wants to run a User Management, there will be a check of the credentials for "admin" and the User Management will be run.

2.3. Management plugins

2.3.1. User Manager

The User Manager is a plugin intended only for the system administrator and the users from the group “Administrators” (see Figure 2, User Manager plugin). The plugin implements fundamental user management functions. An administrator can create new users, search for user/s, edit existing user data and profile and delete Users. The plugin has fields for all the user related information. It has three buttons for the following functions:

1. Add User. The add user button triggers the write user method and can be successful only if all user information is given. If one field is left empty the plugin will show an error massage.
2. Find User. The search procedure can be executed only when a Username is typed. If no such user is found the fields are empty and only the UserID field will show value 0. Meaning that no such user is found.
3. Delete User. The deletion of a user will be successful if the given username is found in the server and is deleted. In all other cases the result will be error.



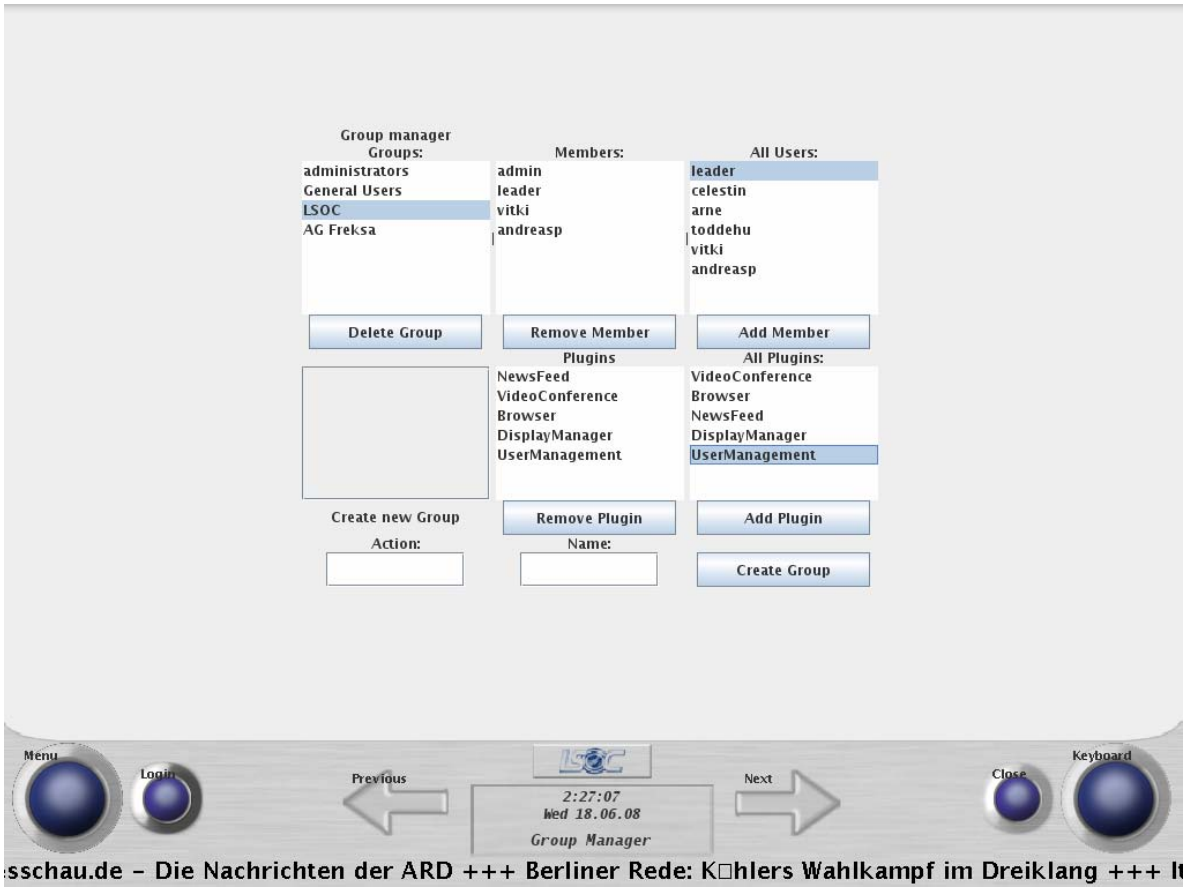
(Figure 2. User Manager plugin)

2.3.2. Group Manager

The Group Manager plugin is divided into five lists. In the first list there are all user groups of the system, in the second list are gathered the members of the selected group. The third list contains all users on the system, next comes the list of the plugins, allowed to be used by the members of the selected group and the fifth and last one is the list of all installed plugins. The Group Manager plugin can be started in two modes depending on the User credentials.

Admin: In the Group Manager plugin the administrator can create and delete groups and also assign and remove user members and plugins to/from the selected group.

User: In this mode the Group Manager plugin fulfills the functions create, edit and delete custom user groups and add or remove members of the group on user level. The user can not see or edit the standard groups. (see Figure 3, Group Manager plugin)



(Figure 3. Group Manager plugin)

2.4. plugins Groups

The plugins are divided into three standard groups and custom groups, defined by the different users. Practically speaking a group in the LDAP directory represents a list of members (uid), which means that one user can be a member of several groups at a time. The plugin groups combine plugins, which can be executed by users, who possess the same credentials. This way it is possible to change the rights of all members together by changing the whole group and it is not necessary to make the change separately on each user.

The three standard groups are as follows:

1. Administrative tools group

The plugins from this group are plugins for administration like the User Manager or the Group Manager plugins.

2. User tools group

That is the second of the standard groups. The plugins are defined by the administrator (by default that means all tools except the Administration ones).

3. Public tools group

The plugins in this group are to use only the publicly accessible plugins, defined by the administrator. The public tools group contains all the plugins that can be accessed by any one including guests. In the guest group all plugins that do not use any personally relevant information are collected.

4. User defined groups

Next to the three standard groups there is the category “user defined groups” which contains all the groups made by the users themselves. These groups can be many and various, depending on the demands and wishes of the users who create them. Any user, who is registered and logged on the system can create custom groups and assign members to them. In this case only the owner/creator of the group and the administrator are allowed to modify the group by adding or removing a user.

3. Dependencies and installation

In this chapter the dependencies (minimal requirements in general) to run the project are described. After that comes the installation of the OpenLDAP server and the developed schema with its properties and dependencies.

3.1. Dependencies

To run the LSOC system the following requirements have to be satisfied:

1. A computer with enough memory and processor for running Java Virtual Machine.
2. About 100MB free space (depends on the number of the users on the system – LDAP and MySQL Database entries)
3. MySQL database running
4. LDAP Server running
5. Network with a connection to the Internet

3.2. Installation

The installation of the OpenLDAP Server on Linux Ubuntu Desktop Edition is rather simple. The installation and configuration can be done in less than 30 minutes by an experienced user (For more information see Addendum chapter 2, OpenLDAP Installation).

3.2.1. Schema

OpenLDAP is distributed with a set of schema specifications for general usage. The general schema that is installed with OpenLDAP is used only for the Distinguished Name (CN), User ID (also known as “uid” or login) and it is extended with the information needed for the project. The LDAP schema was extended to cover the whole user related information needed for the LSOC project.

Attribute Type Definitions

The attribute type definitions are the definitions of the types of data that will be stored. Those are the properties of the object which will stored in the entry.

- lsocLogin
  The attribute is used for the login or the username of the user.
- userPassword
  The attribute is for the password of the user.
- lsocTitle
  The attribute stores the title of the user.
- lsocFirstName
  The attribute stores the first name of the user.
- lsocSurname
  The attribute stores the last name or family name of the user.

Object Class Definitions

The object class definition describes the relation between the attributes. Every object class can have required attributes (example: distinguished name cn) and optional attributes that are not necessary for the entry to be stored.

- lsocAccount
  The class is used for all the user information that has to be stored in the LDAP Directory. It consists of the following attributes:
  - cn
    Distinguished name of the entry in the LDAP Server
  - uid
    user id (in fact the user name or login of the user)
  - uidNumber
    Unique user identifying number in the LDAP Server
  - lsocTitle
    The title of the user. Such as Mr., Ms., Prof., Dr etc.
  - lsocFirstName
    The first name of the user.
  - lsocSurname
    The last name of the user.
  - userPassword
    Password of the user.

- lsocGroup
  The class is about the user groups of the system
  - cn - Distinguished name of the entry in the LDAP Server. This entry must have the same value as the plugin group cn so that resolving the relation between user and plugin groups can be successfully fulfilled.
  - gidNumber – identifing number

- pluginGroup
  The class is about the plugin groups of the system
  - cn - Distinguished name of the entry in the LDAP Server. Like in the previous group, the entry here must be the same value as the user group cn for resolving the relation between the user and plugin groups.
  - gidNumber - identifing number

3.2.2. Database

For the project we use the standard Berkeley database (BDB) from Oracle that comes with the standard installation of the OpenLDAP Server.

4. Data Storage

4.1. Tables and Databases

The connection between the two databases is implemented in the LDAP Handler. The Handler uses only one ID number, auto-generated by the MySQL server. This ID is stored in both databases and is used as a connector between the user information from the MySQL Table and the LDAP server. The process of combining the user information is executed in the LDAP Handler each time when user information is accessed (see Figure 5, Databases).

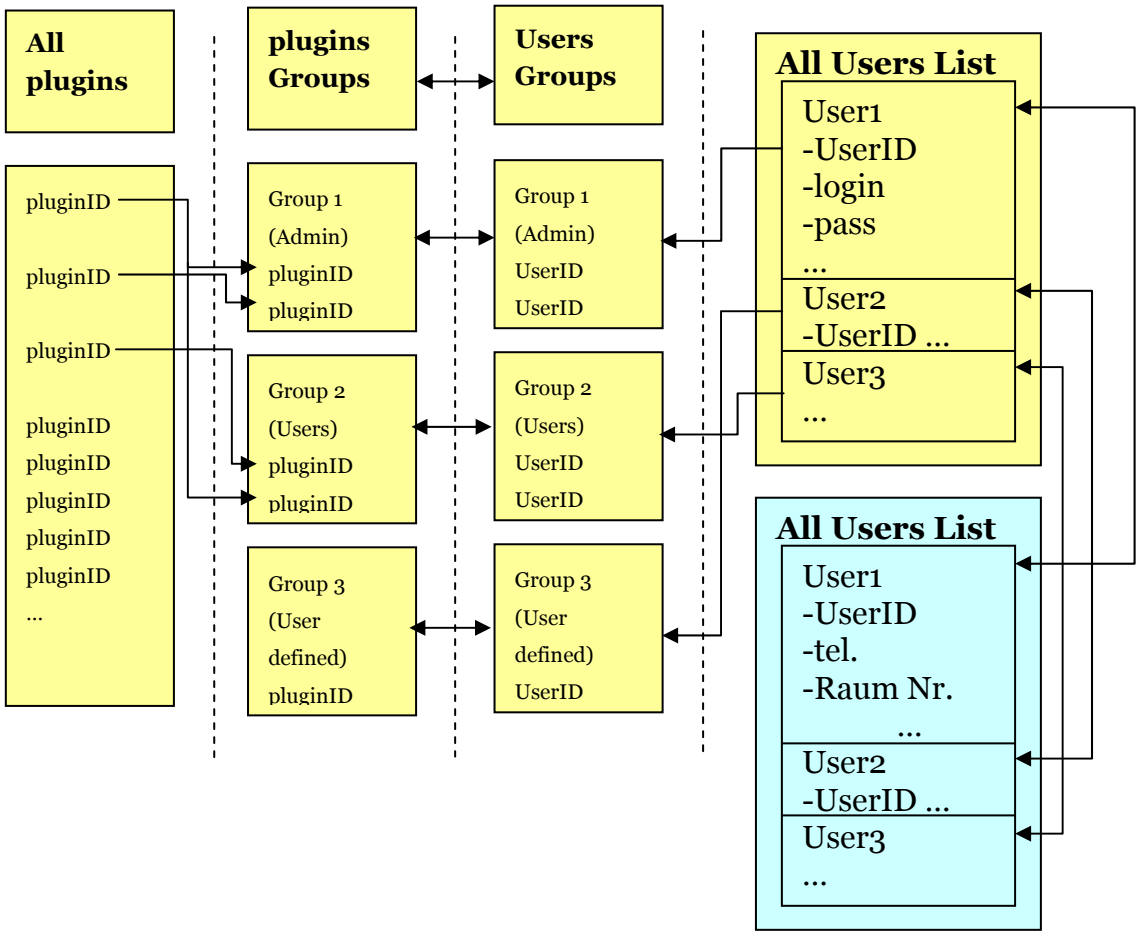
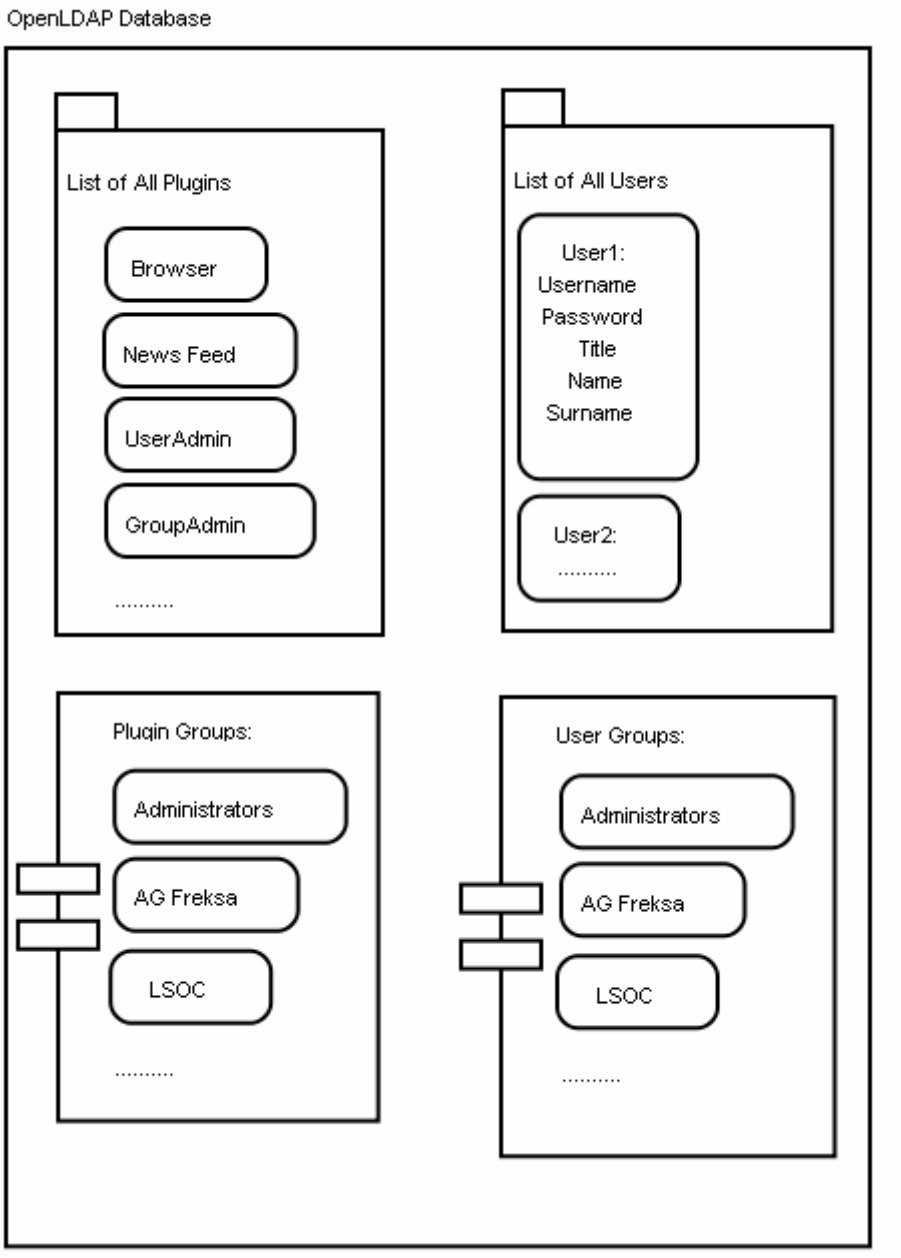


Figure 5. Databases (LDAP in yellow, MySQL in blue)

4.1.1. LDAP Entries

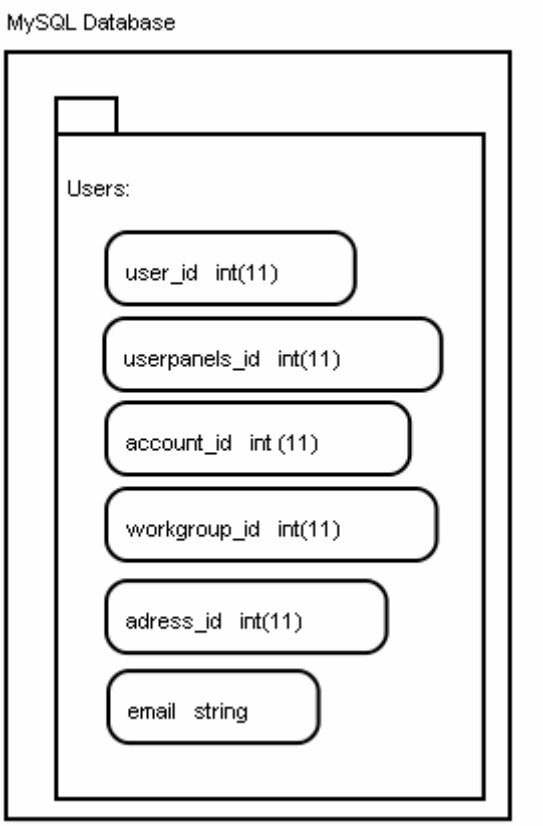
The following diagram describes the entries in the LDAP server. There are four directories. The first one contains all installed plugins in the system (List of All plugins). The second covers all the registered users in the system (List of All Users). The third directory is about the groups of plugins (plugin Groups) and it has the same entries as the fourth, which includes all the user groups (User Groups). The third and the fourth directories are always in synchrony. By creating a new group in the Group Manager plugin, the related entries in both directories are created. The same holds also when deleting a group- the group is deleted in both of the mentioned directories (see Figure 6, Database, LDAP Entries).



(Figure 6. Database, LDAP Entries)

4.1.2. MySQL Entries

The table entries of the MySQL server are not considered private user data. In Figure 7 (Database, MySQL Entries) the initial table entries are shown. The MySQL server is also used for storing other properties and configuration data of the LSOC Project. The table can be easily extended when more user information is needed and it can be changed without the need of any changes in the LDAP user related information.



(Figure 7. Database, MySQL Entries)

5. Summary

The goal of this bachelor report is to provide a detailed description and analysis of the user management for the LSOC project as a part of a smart building concept, based on a network of touch screens with small sized personal computers and a server for storing data. The software has the ability to manage the network and supply the different groups of users (administrators, users and guests) with certain functionality, some of which have more privileges in the system then others. As the project is plugin oriented, the user management is also divided in two plugins- one for managing the users' data and one for managing the groups of users.

The user management of a plugin based system can be accomplished in many different ways. The goal of this work is to create an easy and unpretentious, but effective and resource-flexible solution. The concept uses undemanding OpenLDAP and MySQL servers, which are very popular and easy to use. By combining those two data storing possibilities the high standards for private data holding and processing demanded for in Germany in general and, specifically, at Universität Bremen are ensured.

As the project is intended for embedded computers with limited resources and touch screen interfaces the system and the user management, as a part of the system, have to be small and efficient, but still user friendly. The efficiency is achieved by dividing the implementation of the accesses to the databases in two external classes that work directly with the OpenLDAP and the MySQL server. The Java environment on a Linux based PC provides a very good combination of efficiency and portability. Due to the Java environment the project can also easily be used with Microsoft Windows or Apple's MacOS, or any operating system that has a Java Virtual Machine.

For accomplishing the tasks of the user management fundamental directory access methods are used. To access the OpenLDAP Server the LDAP library developed by Novell is used. As OpenLDAP is based on the X.500 Directory Access Protocol (DAP) it can be easily migrated to other Directory Services for storing the user data.

The main goal of this work is accomplished and provides the functionality needed for the user management for the Location Sensitive Office Communication (LSOC) project.

References

- [LSOCPR] LSOC Project official web site
  http://lsoc.informatik.uni-bremen.de/ (call 01.06.2008)
- [OLDAP] OpenLDAP official web site (call 10.05.2008)
  http://www.openldap.org/
- [NTSCA] Netscape LDAP SDK official site (call 01.05.2008)
  http://www.idevelopment.info/data/Programming/java/ldap/netscape/SUB_netscape_sdk.shtml
- [NOVELL] JLDAP from Novell official site (call 01.05.2008)
  http://www.openldap.org/jldap/
- [Bennett'92] Bennett, J.: Regulating Privacy: Data Protection and Public Policy in Europe and the U.S., Cornell University PressColin 1992, ISBN:0801480108
  Web link:
  http://books.google.com/books?hl=en&lr=&id=D17pcdORb9UC&oi=fnd&pg=PR7&dq=data+protection+definition&ots=iqqlVAzsie&sig=rbYWxC6-hC_ru-0XOC9IQRhcfq4#PPA34,M1 (call 01.06.2008)

- [ISO/IEC] ISO/IEC 27002:2005 Information technology -- Security techniques - Code of Practice for Information Security Management
  http://iso27001security.com/html/27002.html (call 01.06.2008)
- [DataSec] Data Security in University of Bremen (call 01.06.2008)
- [Flaherty'92], David H.: Protecting Privacy in Surveillance Societies, The University of North Carolina Press, Chapel Hill and London, 1992
- [Kubicek] Kubicek, Herbert: Datenschutz I: Rechtliche Grundlagen und deren technisch-organisatorische Umsetzung, Arbeits Gruppe Informationsmanagement
  Web link (call 28.06.2008):
  http://www.agim.informatik.uni-bremen.de/agim/sixcms/media.php/35/datenschutz_vorl_2005_11_17.pdfhttps://elearning.uni-bremen.de/zmml/datenschutz_l.pdf
- [DSE] Datenschutzerklärung für Studierende an der Univerität Bremen
  Web link (call 28.06.2008):
  https://elearning.uni-bremen.de/zmml/datenschutz_s.pdf
- [ORACLE] Oracle official homepage (call 30.05.2008)
  http://www.oracle.com/technology/products/berkeley-db/index.html
- [MySQL] Sun MySQL official homepage (call 30.05.2008):
  http://www.mysql.com/

Addendum

1. Hardware

1.1. PC Mini ITX Specifications

A custom small sized PC Mini ITX with Mainboard VIA CX700M Chipset. Processor 1GHz VIA C7 ultra-low CPU-utilization. Memory 1GB 240-pin DDR2 DIMM. Integrated UniChrome Pro II 3D/2D Graphic and Video Processor.

For more information see the manufacturer site (call 01.06.2008):
http://www.tragant.de/

1.2. Touch screen specifications

Chi Mei G121S1-L01 (CH-01-004)

The G121S1-L01 model is a 12.1" TFT-LCD module with a 2-CCFL Backlight Unit and a 20-pin 1ch-LVDS interface. This module supports 800 x 600 SVGA mode and displays 262,144 colors. The inverter module for the Backlight Unit is not built in.

For more technical information please visit the official web site of DISTEC GmbH (call 15.05.2008) http://www.distec.de/,
direct link to Display specifications (call 15.05.2008):
http://www.distec.de/de/Produkte%5CTFTDisplays%5CG121S1-L01.html

2. OpenLDAP Installation

Following the instructions from the Ubuntu forum (http://wiki.ubuntu-forum.de/index.php/OpenLDAP, call 01.06.2008) the installation process goes through the following steps:

1. Download the LDAP libraries and the slapd deamon (slapd ldap-utils)
   Optional can be installed: nmap php5-ldap db4.2-util for easier configuration.
2. Install the selected packets.
3. Set administrator password with commando: slappasswd
4. Set the Suffix in the slapd.conf (Example: "dc=lsoc,dc=local")
   The suffix is needed to access the LDAP server. It shows the URL of the server and is needed also for login the administrator of the server.
5. Set the administrator password and login.
   (Example: rootdn "cn=admin,dc=meinedomain,dc=local"
   rootpw {SSHA}4GMPGS/UQTOJ7LdI+iOu7lExQAbpzX6/)
6. Restart the LDAP Deamon slapd with commando:
   /etc/init.d/slapd restart
7. Set settings in the LDAP Configuration (ldap.conf):
   Example:
   ldap_version 3
   URI ldap://192.168.x.x:389

SIZELIMIT 0
TIMELIMIT 0
DEREF never
BASE dc=meinedomain, dc=local

8. Create the first entry of the administrator in the LDAP Directory.
   Example: create a file base.ldif with the following lines:
   dn:dc=meinedomain,dc=local
   objectClass: dcObject
   objectClass: organization
   o: meinedomain
   dc: meinedomain

   dn:cn=admin,dc=meinedomain,dc=local
   objectClass: organizationalRole
   cn: admin
9. Add the entry in the database
   Example:
   ldapadd -x -W -D cn=admin,dc=meinedomain,dc=local -f base.ldif
   Enter LDAP Password: xxxxxx
10. Test the server with the command: ldapsearch –x
    It shows the new admin entry with some internal information.

For more information and help see (call 01.06.08): http://wiki.ubuntu-forum.de/index.php/OpenLDAP

3. User Management Interface

The description of the programmatic interface of the user management related to the other parts/plugins of the LSOC system is shown in the following.

Needed Functions:
1. Authorize a user to run a plugin.
2. Read user object (all user information)
3. Read user status – logged in or not.
4. Read user group/groups (contained in the LDAP server not in the user object)
5. Read users group
6. Read user owned group/groups
7. Read user owned PanelID

3.1. API Internal Interfaces

The next paragraphs present the description of the internal interface of the user management.

Reading user information and status

- Package: level0.io.LDAPHandler.java

public User readUser(User _fromDB)
//Reads all user data from the database and returns object from type User.

Writing user information

- Package: level0.core.Authorize.java

public boolean writeUser(User userToWrite)
//Writes a user in the database. Returns true if succeeded.

public boolean deleteUser(User userToDelete)
//Deletes a user from the database. Returns true if succeeded.

public boolean updateUser(User _upToDate)
//Change a user information in the database. Returns true if succeeded.

Reading user credentials (Rights)

- Package: level0.core.Authorize.java

public boolean Auth(int UserID, int pluginID)
// Reads user credentials from the database. Returns true if the user with UserID have permissions to run the plugin with pluginID.

Writing user credentials (Rights)

Writing the user credentials will be realized with the help of the Write user information methods (ee. Writing user information).

Internal procedures

Here are the Methods used for manipulating internal user data in the LDAP and status information such as Logged User.

Login and Logout

- Package: levelo.core.Authorise.java

public boolean Login(String User, String Pass)
//Changes the status of the logged user on the system. Return true to the system to change the visible sign that the user is logged on.

public boolean Logout(String User)
//Changes the status of the logged user on the system. Return true to the system to change the visible sign that the user is logged out.

- Package: levelo.io.LDAPHandler.java

protected String LoggedUser="Guest";
//This String variable is used to save the
public String getpluginGroup(String pluginID)
//Returns the name of the group into which the plugin with pluginID is found.

public List readGroups()
//Read the Groups List

public List readUsers()
//Read the Users List.

public List readMembers(String Group)
//Reads the members of a group

public List readAllplugins()
//Reads all plugins into a list

public boolean checkpluginGroup(String pluginName, String pluginGroup)
//Checks if a plugin is in a specific group

public List readplugins(String Group)
//Reads the plugin members of a group

public boolean checkUser(User userToCheck)
//This Method checks if a user name is avalaible on the system or is already taken.
public String getGroup(String UserID)
//Return the group of a user.

public String getpluginGroup(String pluginID)
//Returns the plugin group of a plugin.

public List readGroups()
// Returns a list with the names of all plugins

public List readUsers()
//Returns a list with the names of all Users

public List readMembers(String Group)
//Returns a list with the names of the members of a group

public List readAllplugins()
//Returns the list with the names of all plugins installed on the system

public boolean addplugin()
//Adds new plugin in the administrators group. After that the administrator can decide in which group to place the plugin

public boolean removeplugin()
//Removes a plugin from the Server

public boolean createGroup()
//Creates new user group and the according plugin group on the server.

public boolean addpluginMember(String pluginName, String toGroup)
//Adds plugin Member to the selected Group in User Groups

public boolean addMember (String Member, String Group)
//Add memeber to selected group in the LDAP Server.

public boolean removeMember (String Member, String Group)
//Add memeber to selected group in the LDAP Server.

public boolean removepluginMember(String pluginName, String Group)
//Remove plugin memeber from selected group in the LDAP Server.

public boolean createGroup(String groupName)
//create user group and the belonging to it plugin group in the LDAP Server.

public boolean deleteGroup(String groupName)
//Deletes a user group and the belonging to it plugin group in the LDAP Server.