

## Project Task: Design a directory for handling of person names, phone numbers and e-mail addresses.

A directory is a kind of database optimized for efficient reading and searching but with lower requirements on updates. It is a common tool used to organize information of various kinds. You have been exposed to many directories. To get an idea what a - laptop computer works and handles names, phone numbers and e-mail addresses etc.

A directory works can be seen as a server used by two kinds of clients:

- 1) the ordinary users of directory
- 2) the administrator of the directory

Ordinary users are only allowed to perform a restricted set of operations, while the administrators have full authority to make all operations possible on the directory.

The data that we store in a directory is intended to be well organized and represent important information and information structures. To understand those structures we need a model or a so called schema as a guide over the information and its structures. In object oriented terminology you make design of the directory built by use of class inheritance and class composition structures defining the attributes types of the classes involved.

Directories are typically organized hierarchically in tree structures. The nodes in such a tree are objects distinguished by their names and attributes. For sure identification of an object in a tree, different forms of unique names or identifiers can be used. Examples of such unique identifiers are domain names, telephone numbers (complete with area and country code) but also automatically generated ID numbers such as Universally Unique Identifier (UUID) commonly used to identify objects, messages and sessions in a distributed system without the need for coordination to avoid naming conflicts.

Since your studies are related to computer networks and protocols you are supposed to use terminology used by a protocol called LDAP used for network distributed directories. By studying what LDAP is, you can indirectly get an understanding what a directory is. Another way to get a picture of what this task is about is to take a look at an open source implementation of a directory server implemented in Java, for example OpenDS.

In network distributed applications the operations to be performed on the directory must be possible to do remotely from any of the above kinds of clients by sending messages to the server over a computer network. For this purpose the light directory access protocol (LDAP) has been designed, the current version is called v3 and is defined in an RFC called 2251 and its update 4511 (see also 4510 for an overview of related protocols). To get information about LDAP and its history you can also read Wikipedia and other popular descriptions of LDAP and about directory structures. All this you can easily find out about by using Google or other search engine for a while.

A directory intended to be accessed by use of the LDAP protocol must have some specific properties fulfilled. For example, all entries stored in the directory must have a

unique "Distinguished Name," (DN) composed of two parts: the Relative Distinguished Name (RDN) and the location within the LDAP directory, i.e. where it is placed in the hierarchical, directory tree, structure. Each actual operation to an LDAP directory, doing something with the data in the directory, must be part of a session starting with the setup of a secure TLS connection, using the operation *StartTLS* over which the client also is authenticated using the operation *Bind*. Then when a secure session is established "ordinary" operations such as *Search and Compare*, *Update Data*, *Abandon* etc can be made. Such operation messages need to be associated with a unique message ID since answers are allowed to be delayed and can come in any order from the server or chain of servers involved in answering. The secure connection, carried by a TCP/IP session, is finally turned down by use of an *Unbind* operation.

Your task is to make an implementation of a directory limited to its data handling parts that can be accessed and operated upon via the two types of clients mentioned above. To test the function of the directory as a server it can be useful to implement simple version of the clients and especially their APIs as well.

To be able to solve the task and get full credit you must:

1. Gather information about directories and directory structures that can be accessed via LDAP. Create a requirement specification that tells what you need to fulfill.
2. Make an object oriented design that is detailed enough to work as an implementation proposal. From this design proposal it shall be possible to divide and share the implementation work in the project group.
3. Create implementation classes and a main program that builds an empty directory that can be filled with directory entries.
4. Test the result using a set of operations that covers all functionality. Start with operations that create entries in the directory and then do different search, read, compare and update operations.
5. Explain and demonstrate your design, implementation and its function by use of test cases, and be ready to modify this if asked for, to the examiner.

The project task has a mandatory core that all groups must do including the functional requirements on actual directory reading and updating related operations. An ambitious group can however do extra work on one of the possible extensions below:

1. data structures and algorithms that gives good performance
2. client tools (graphical or textual) for ordinary user and for administrator benefit
3. making the directory "network attached" by adding LDAP functions for setup, binding and closure of secure sessions over a network etc.
4. making a RMI based access to the directory
5. making your solution compliant with JNDI, an API providing naming and directory functionality to applications written in Java.