A Web-based Corporate Directory Application Using LDAP (V3) (RFC 2251)

Submitted by: Amol Gaikaiwari
Computer Science Department
University of Illinois at Springfield
One University Plaza
Springfield, IL 62703-5407
agaik01s@uis.edu

Advisor: Dr. Kamyar Dezhgosha
University of Illinois at Springfield
One University Plaza
Springfield, IL 62703-5407
kdezh1@uis.edu

Abstract

The main objectives of this project are:

- Developing an understanding of the evolution of the Directory Access Protocol (DAP)
- Acquiring LDAP specific know-how about installation of LDAP server, designing Directory Information Trees (DITs), setting up Access Control Lists (ACLs), optimizing searching of directory data etc.

The project implements a system, which utilizes and demonstrates the features/ benefits of the LDAP protocol. The system stores data using DITs on the LDAP server and it employs security policies using ACLs. The users of this online directory application are authenticated using the ACLs.

The system lets users query the LDAP server for accessing information about people, mailing lists and devices available within the enterprise. It also allows users to upload binary data such as photographs and audio files to the LDAP server. This system demonstrates the advantages offered by the Single Sign on (SSO), a concept that is rapidly gaining popularity in the industry.

Key Words: Light-weight Directory Access Protocol (LDAP), OpenLDAP, slapd, Servlets, Berkley database (BDB), Application Programming Interface (API), LDAP data interchange format (LDIF), Simple Authentication Security Layer (SASL)

1 Introduction

A directory is an essential part of people's lives today. By accessing static directories like the paper phone books provided by phone companies, people can easily find out phone numbers of other people, shops, organizations etc.

All information in the paper directories is static and does not change in real time. Also, there is no means to authenticate the information of the directory user before granting him access to thousands of phone numbers/postal addresses etc. Hence, the information contained in static directories is not secure.

By contrast, online directories (also referred to as dynamic directories) have the capacity to be much more up-to-date. However, a point to note here is that the frequency with which these directories can be updated depends solely on the directory administrator (Zacker 1997). But more than often, these directories are pretty much updated to reflect accurate information more frequently than static directories.

Dynamic directories have a distinct advantage over their static peers in the sense that information can be organized in many different ways in a dynamic directory. More amount of information can be added to an online directory like the organization, department in which that person works or the location of his/her office etc. Online directories are also more flexible and dynamic.

The information in online directories can be secured too. This enables enterprises to restrict access to information based on different criteria thereby protecting confidential information from being accessible to unintended audiences.

Online directories also enable information to be centralized facilitating the creation of a central repository of information, thereby reducing the possibility of redundancy and inaccurate information. Centralized systems also help improve the security for enterprises because accesses to different systems can be set up centrally which vastly improves monitoring security. This centralization, alternatively referenced as single sign on (SSO), can be easily implemented using the LDAP protocol.

The superior update capacity of online directories not only tends to keep information more up-to-date; it also can be used to distribute the update responsibility. The closer information is to its source, the more accurate and timely the information is likely to be (Weltman and Dahbura, 2000). There are at least three reasons for this:

- The source of the information is, by definition, the most accurate.
- Extra delay and opportunity for error between the source and the directory are eliminated if the source makes the update itself.
- Depending on the information and the application, the source is likely to be the party most motivated to maintain the information correctly.

An online directory can be extended without the need for a redesign. Data can be organized in various different ways and can be optimized for searching based on many different criteria.

Overall, online directories have innumerable advantages over static directories. There are standards defined for creating and accessing online directories and the Lightweight Directory Access Protocol (LDAP) is one such industry standard which has grown very popular throughout the industry.

2 Evolution of Online Directories

In the late 70s and early 80s, the International Telecommunication Union (ITU) (www.itu.int) started work on the X.400 series of email standards. This email standard required a directory of names (and other information) that could be accessed across networks in a hierarchical fashion (similar to Domain Name Service) (Carter, 2003).

This need for a global network based directory led the ITU to develop the X.500 series of standards and specifically X.519, which defined Directory Access Protocol - the protocol for accessing a networked directory service.

According to the X500 standard, a directory service should include the following 5 elements (Zacker, 1997).

- A directory service has a model that defines the basic structure in which information is stored.
- A directory service has a tree-like hierarchy, called a directory information tree.
- A directory service has a collection of command functions that clients can use to manipulate directory entries.
- A directory service has a system to authenticate users.
- A directory service has a distribution model that lets clients (called directory user agents) view the data stored on multiple servers (called directory system agents) as a single entity.

The X.400 and X.500 series of standards came bundled with the whole OSI stack and consumed serious resources.

Developing client applications on such a protocol heavy standard made the client applications that communicated with the X500 server to be very complex. There were a very wide variety of options that needed to be configured before the client applications could work with an X500 server. But since most clients ran in a TCP/IP network environment and needed only a very small subset of these options, work began on a lightweight client access protocol. The work was successful and then, all commercial X.500 servers came with an LDAP gateway (Figure 1), a program that translates between the lightweight, TCP/IP-based client protocol and the native X.500 protocol of the server.

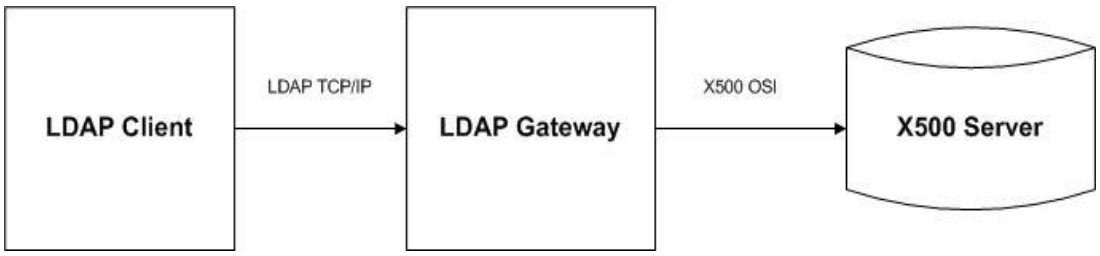

Figure 1: LDAP gateway for a X500 Server

The first RFC for LDAP (RFC 1487) was published in 1993, which described the LDAP gateway to X500 servers and was titled as "X.500 Lightweight Access". It was the second version of the LDAP specification, which turned out to be very popular (RFC 1777 "Lightweight Directory Access Protocol"). The authors of the LDAP protocol designed and developed a standard SDK for application developers. They also defined client API for accessing the protocol. The initial LDAP specifications excluded definitions for data replication, referrals and chaining.

The LDAP V3 (RFC 2251) implements these specifications (Wahl, Howles and Kille, 1997). Research is currently being done on standardization of more X500 features that are not yet a part of LDAP.

LDAP differs from Directory Access Protocol (DAP) used by X500 mainly in the following respects (Weltman and Dahlbura, 2000):

- LDAP generally uses TCP/IP (though it can be used with other protocols too) whereas DAP uses OSI transport/network layers.
- LDAP offers lesser features as compared to DAP. It excludes most of the very rarely used DAP features.
- DAP follows a top-down approach and divides regions by a top-level node divided by country. All companies are then built at lower levels of the directory information tree. This may not necessarily work because often, companies can span multiple countries. With LDAP, directories can be designed using a bottom-up approach to meet the actual needs of an organization. Different organizational units can then appear as a single organization by setting up referrals

LDAP is a protocol that defines the method by which directory data is accessed. Secondly, it also defines and describes how data is represented in the directory service (the data model). Finally, it also describes how data is imported and exported from the directory service via the LDAP Data Interchange Format (LDIF). Figure 2 illustrates the scope of the LDAP protocol.

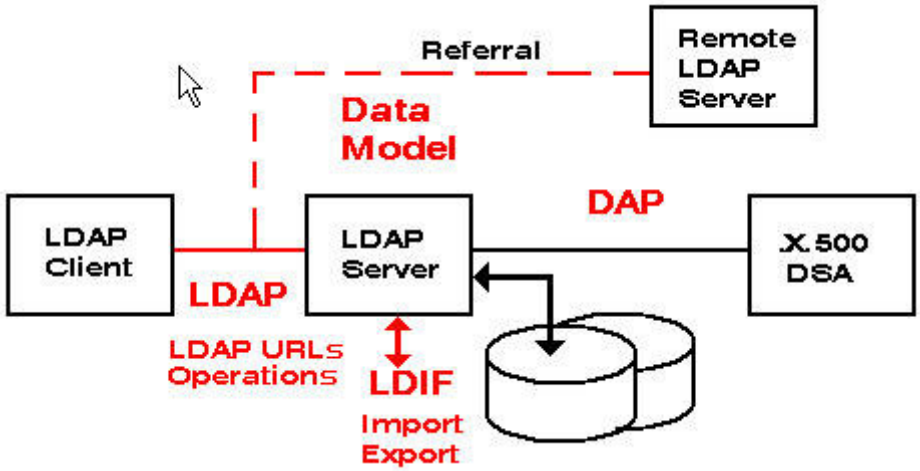

Figure 2: Scope of LDAP Protocol
(Source: LDAP for Rocket Scientists)

3 Organization of LDAP Directories

3.1 LDAP Information Model

Data is represented in an LDAP enabled directory as a hierarchy of objects called entries. The resulting tree structure is called as a Directory Information Tree (DIT). The top of the tree is called as the base/suffix.

Each entry can have only one parent and multiple child entries. All child entries for one parent are at the same level and are called as siblings. Each entry in the directory can be uniquely identified using a Distinguished Name (DN). There can be more attributes, which describe the entry.

Each entry must at least have the attribute “ObjectClass” which defines the entry. The object class defines which attributes are “required” by the entry and those which can be optionally included (Wahl, Howles and Kille, 1997).

The entire LDAP directory data is organized in terms of object classes and attributes. Figure 3 illustrates the use of object classes and attributes in the DIT, as well as the hierarchy of a typical DIT.

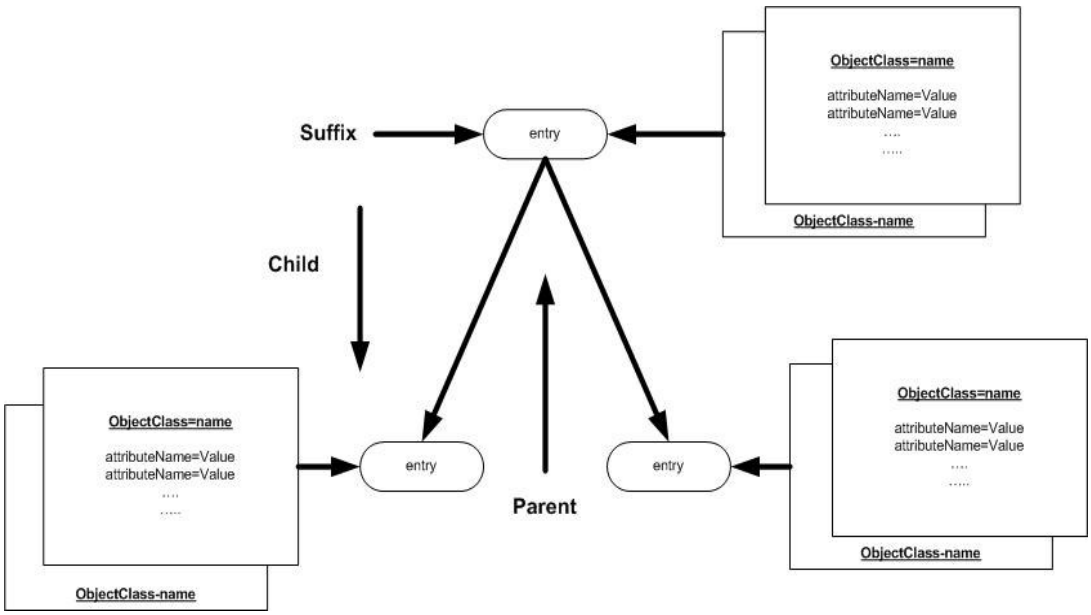

Figure 3: Hierarchy of a typical DIT

All object classes and their attributes are defined in schemas (packaging units) and are reusable. All the schemas, which include the object classes and attributes in a typical LDAP implementation, should be known to the LDAP server. The methods for configuring servers to include the schemas differ with implementation. Attributes and object classes can "cross" schemas i.e. an attribute defined in a schema A can be used by an object class defined in schema B.

3.2 LDAP Naming Model

Figure 3 above illustrated that entries in LDAP are organized into a hierarchical tree like structure and that each entry has a "distinguished name" by which it can be identified uniquely in the directory information tree (DIT). In this structure, at the top, there is a root node called as suffix or naming context. Underneath, there can be multiple child nodes.

Each sibling can be distinguished from another by a relative distinguished name (RDN). This name is unique to that child at the same level in the hierarchy. This RDN consists of the attribute name and its value (for e.g. o=company). A DN can then be built by taking the RDN of the entry and qualifying it by RDN of each parent until the suffix is reached (Hodges and Morgan, 2002).

3.3 LDAP Functional Model

This model describes the operations that can be performed on the information contained in the DIT. There are three operations that can be performed.

- **Search and Read:** The read operation retrieves the attributes of an entry whose name is known. The list operation enumerates the children of a given entry. The search

operation selects entries from a defined area of the tree based on some selection criteria known as a search filter. For each matching entry, a requested set of attributes (with or without values) is returned. The searched entries can span a single entry, an entry's children, or an entire sub-tree

- **Modify:** This facilitates adding, deleting and modifying information contained within entries as well as the entries themselves.
- **Authenticate:** This facilitates the "bind/unbind" operations, which allow a user to access the information contained in the DIT.

3.4 LDAP Security Model

This model allows information in the directory to be secured. It encourages data integrity and privacy by authenticating users, allowing the root administrator of the DIT to set up password policies to allow/disallow users from accessing classified information, etc (Hodges and Morgan, 2002).

4 LDAP Implementations in the industry

Following are some of the widely used LDAP implementations in the industry

4.1 IBM Tivoli Directory Server

IBM's Tivoli Directory Server is a powerful, security-rich and standards-compliant enterprise directory for corporate intranets and the Internet. The users can choose authentication strategies like simple user ID and password authentication, or the more secure digital certificate-based authentication structure. The Directory Server also includes a Simple Authentication Security Layer (SASL) plug-in interface like Kerberos authentication.

The access control features in IBM Tivoli Directory Server extend to the attribute level, enabling self service and delegated administration while also offering protection of access control list (ACL) values within the directory, preventing unauthorized users from changing the security assigned to objects within the directory (IBM Tivoli Directory Server, 2004).

4.2 Microsoft Active Directory (AD)

Microsoft's AD provides a directory service designed for distributed networking environments. AD lets organizations efficiently share and manage information about network resources and users. In addition, since Active Directory is integrated with the server versions of Windows OS, it acts as the central authority for network security, letting the operating system readily verify a user's identity and control his/her access to network resources.

Combined, these capabilities let organizations apply standardized business rules to distributed applications and network resources, without requiring administrators to maintain a variety of specialized directories (Active Directory Overview, 1999).

4.3 Netscape Directory Server

Netscape Directory Server provides a comprehensive, enterprise-wide directory service for managing information about users, groups, and access control lists. Netscape Directory Server 4.0 supports versions 2 and 3 of the Lightweight Directory Access Protocol (LDAP). It provides the capability of letting clients across multiple platforms access NDS through its proprietary Java SDK (Netscape Directory Server Release Notes, 2004).

4.4 Oracle Internet Directory

Oracle Internet Directory is an LDAP v3 service that offers the flexibility and extensibility of LDAP along with the scalability and reliability of the Oracle9i platform. The Oracle Internet Directory server is implemented as an application running on the Oracle9i Database. Through its tight integration, Oracle Internet Directory effectively leverages the features of the Oracle platform to make it the compelling choice for mission-critical applications (Oracle Internet Directory Service 2002).

4.5 OpenLDAP

The OpenLDAP Project is a collaborative open source effort to develop a robust, commercial-grade, fully featured and open source LDAP suite of applications and development tools. The OpenLDAP service, slapd, is an LDAP directory server that runs on many different platforms. The directory can contain pretty much anything in it. The user may connect it to the global LDAP directory service, or run a service all by itself.

The slapd service implements version 3 of Lightweight Directory Access Protocol. It supports LDAP over both IPv4 and IPv6 and Unix IPC.

Slapd provides a rich and powerful access control facility, allowing the user to control access to the information in the database(s). The user can control access to entries based on LDAP authorization information, IP address, domain name and other criteria. Slapd supports both static and dynamic access control information.

Slapd supports backend databases such as Berkley Database (BDB - www.sleepycat.com), a high-performance transactional database backend and LDBM, a lightweight DBM-based backend.

Multiple database instances: slapd can be configured to serve multiple databases at the same time. This means that a single slapd server can respond to requests for many logically different portions of the LDAP tree, using the same or different databases. Slapd

is highly configurable through a single configuration file called <slapd.conf> that resides on the LDAP server (OpenLDAP 2.2 Administrator's Guide, 2004).

5 Architecture of the LDAP Directory Service Application

The Application is based on 5-tiered model. The Enterprise architecture for the LDAP Directory Service application is shown in Figure 4.

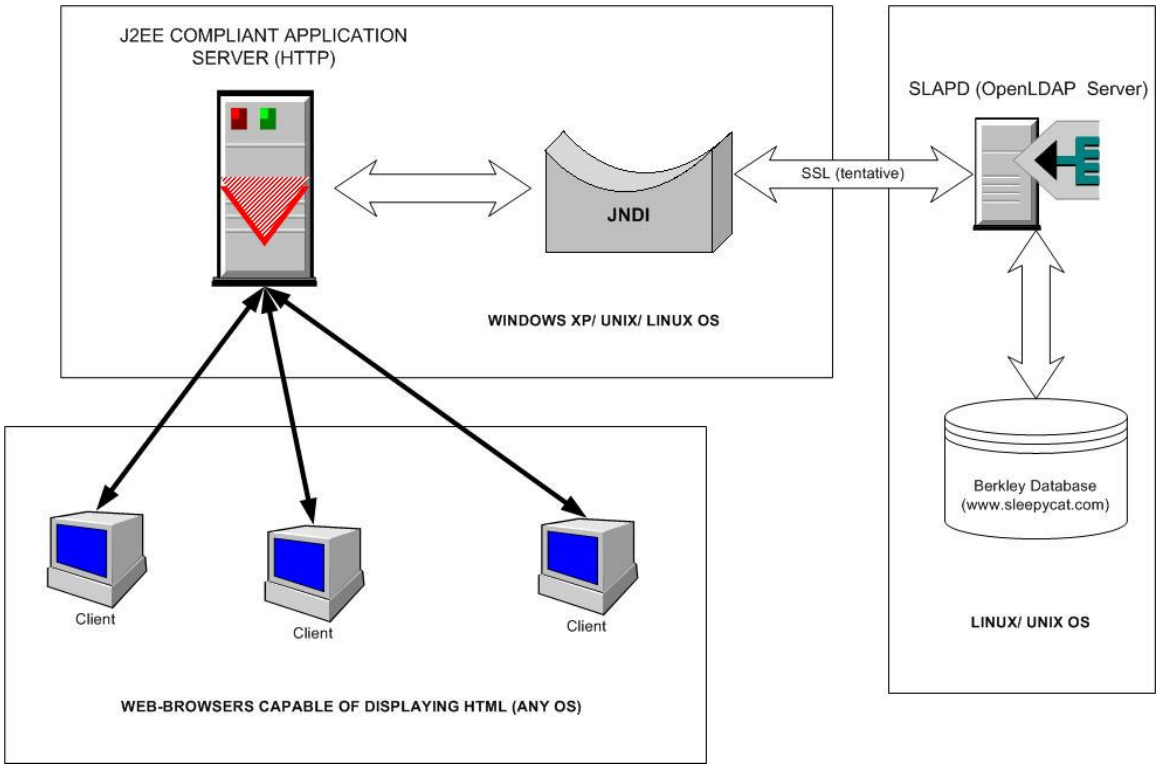

Figure 4: Overall Architecture for the Corporate Directory Application

The 5-tiered architecture shown above has the following major components:

1. **Client:** The client for this application will be a simple web-browser capable of displaying HTML and graphics.
2. **Application Server:** This runs the core of the business logic implemented in the application. This layer handles the process of accepting requests from the client, performing server-side processing and returning the results of the request back to the customer. This Layer is responsible for session management and data persistence and also acts as the presentation layer for the application.
3. **JNDI Bridge:** This layer transfers data to-and-fro between the LDAP server and its Java-based client (written using Java Servlets) that resides on the application server. This layer is responsible for translating the Java-based requests into LDAP protocol specific commands and performing the translated operations on the LDAP server. It is also responsible for translating the results returned by the LDAP server back into an Object-form that can be interpreted by the Java client.

4. **LDAP Server:** This layer manages all the information for this application. It is also responsible for enforcing authentication mechanisms to facilitate restricted data access.
5. **Database**: This layer actually stores the data used in this application.

The entire application is divided into 7 different modules. This division has been performed on the basis of the functionality provided by these modules and the entities on which these operations will be performed. Overall, there are three major entities in this application

1. People
2. Email Lists (user groups)
3. Devices

There are four types of operations that can be performed on these three entities that constitute the 4 major subsystems.

1. Add
2. Remove
3. Modify
4. Search

In addition to these 4 modules, there are 3 additional modules that

1. Authenticate the user based on identification information provided by the user.
2. Provide the user with the capability of uploading binary data such as an image file or an audio file to the web server.
3. Provide an interface for the application to communicate with the LDAP server.

6 Software Development Methodology

6.1 Analysis

Table 1 lists the functionalities included in the web-based Corporate Directory Application using LDAP as well as certain features that will not be supported.

Sr. No.	Feature	Support
1	Corporate Directory Design	Yes
2	Database Format	Berkley Database (BDB)
3	LDAP Implementation	OpenLDAP
4	Platform for LDAP Server	Linux
5	Platform for J2EE Application server	Windows XP
6	OpenLDAP installation and configuration on Linux	Yes
7	Berkley Database Installation on Linux	Yes
8	OpenSSL installation on LINUX	Yes

9	Initial Corporate Directory Database Implementation via Lightweight Data Interchange Format File (LDIF) and slapd configuration file (slapd.conf)	Yes
10	Implementation of data security and authentication via access control lists (ACLs) in the slapd.conf file	Yes
11	Implementation of secure socket layer for data transport between Http Server and slapd	No
12	Design and Development of Corporate Directory Client using Java Servlets and Java Naming and Directory Interface (JNDI)	Yes
13	Demonstrate Database Replication via SLURPD	No
14	Implementation of Referrals and Chaining processes	No
15	Implementation of Kerberos SASL authentication	No
16	Integration with other applications	No
17	Ability to create/modify/delete entries from LDAP server via client application	Yes
18	Ability to search information from LDAP server via client application	Yes
19	Online Help for the user	Yes

Table 1: Proposed Functionalities

6.2 Application Design

The Online Directory Service Application project is divided into 7 modules namely:
1. BindUser Module: Authenticate user and create session
2. AddData Module: Add People/Email Lists/ Devices
3. RemoveData Module: Remove People/Email Lists/ Devices
4. ModifyData Module: Modify People/Email Lists/ Devices
5. SearchData Module: Search People/Email Lists/ Devices
6. UploadData Module: Upload Binary (image/audio) file
7. LDAPBridge Module: Provide methods to interact with the LDAP server

The security is implemented on the LDAP server and so, none of the modules listed implement the security aspect of the add operation. For example, if a user is not authorized to add data to the directory, then, an appropriate error is displayed back to the user if he/she tries to add data to the directory.

6.2.1 BindUser

This module authenticates the directory service application user by *binding* the user to the directory. It accepts the user id and password from the user and tries to authenticate the user based on the provided information. If the credentials match, the user is bound to the LDAP server and a session is created for the user. Using this session, the user can perform different operations on the online directory

6.2.2 AddData Module

This module allows the user to add data to the online directory. The various types of entries that the user can create in the online directory are people, email lists (user groups) and devices.

6.2.3 RemoveData

This module allows the user to remove data from the online directory. The various types of entries that the user can delete from the online directory are people, email lists (user groups) and devices.

6.2.4 ModifyData

This module allows the user to modify data from the online directory. The various types of entries that the user can modify from the online directory are people, email lists (user groups) and devices.

6.2.5 SearchData

This module allows the user to search data from the online directory. The various types of entries that the user can search from the online directory are people, email lists (user groups) and devices.

6.2.6 UploadData

This module allows the user to upload his/her image or audio file into the online directory. This data can then be used with image or voice authentication software for identity verification.

6.2.7 LDAPBridge

This module provides LDAP specific JAVA methods to the remaining 6 modules. The purpose of this module is to consolidate all LDAP specific operations into a single module to increase reusability and maintainability of the application. This also helps in the separation of the JAVA-based client processing from the actual LDAP operations. This module uses JNDI API to access the LDAP server.

6.3 Directory Design

The LDAP protocol specifies the method for importing/exporting or representing data but not the method for storing/manipulating the data. So, the data may actually be stored in a relational database but when it is accessed through the LDAP protocol, it is conceptualized in a Directory Information Tree (DIT). The DIT design for this application is illustrated in Figure 5.

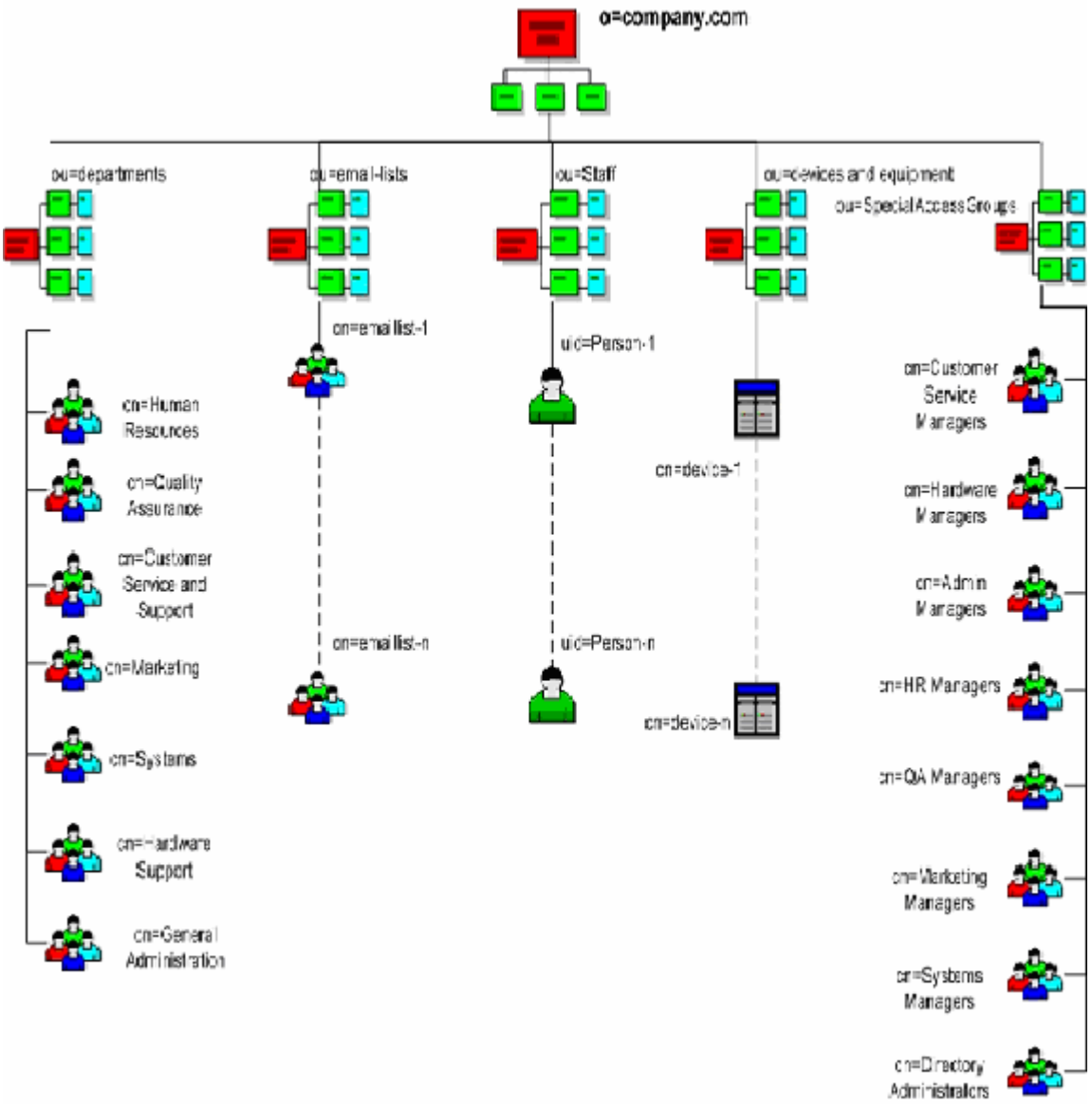

Figure 5: Proposed Directory Design

6.4 Implementation

The application is developed using Java Servlet API, JNDI API and OpenLDAP server. As illustrated in the hi-level architecture diagram in Figure 4, the OpenLDAP server runs on a LINUX OS capable of running the OpenLDAP V3 server implementation. The mid-tier and presentation layer comprising of Servlet and JNDI components runs on Windows XP.

As of now the processor and RAM requirements for the application are as follows

- Any Windows XP/ LINUX compatible processor with minimum processing speed > 1.0 GHz (processors with speeds > 2.0 GHz are preferred though)
- RAM: at least 256 MB
- OS: Windows XP/ LINUX (RedHat 9.0/ Fedora Core Release 1 should work)
- Hard Drive Space required for the entire application to reside (@ 500 MB)

The users of the application will not be required to store anything on their hard drives. The only requirement for the users is that they should have a machine that has a web-browser capable of interpreting HTML, and they should reside in the same network as the server.

7 Conclusion

Online directory applications can be developed using Enterprise Java Technology and the lightweight directory access protocol (LDAP). J2EE, as a distributed component architecture, is the right solution in developing online directory applications due to its write once, run anywhere (WORA) and write once, deploy anywhere (WODA) philosophy. Since the JNDI technology is LDAP protocol based, applications developed using this API can run on any J2EE-compliant web server, on any operating system as well as with any LDAP-compliant directory server. The LDAP protocol is very powerful, extensible and can be used as a very effective security and data storage tool for less volatile data.

8 References

1. Gerald Carter. LDAP System Administration. OReilly Publications, 2003.
2. IBM Tivoli Directory Server. 2004.
   http://www-306.ibm.com/software/tivoli/products/directory-server/
   (10/04/2004)
3. J. Hodges and R. Morgan. "Lightweight Directory Access Protocol (v3):Technical Specification". IETF RFC 3377. 2002.
   http://www.ietf.org/rfc/rfc3377.txt
   (09/26/2004)
4. "LDAP for Rocket Scientists". Zytrax.com - Open Source Guides - for Rocket Scientists. 5 Oct 2004.
   http://www.zytrax.com/books/ldap/
   (10/08/2004)
5. Netscape Directory Server Release Notes. Version: 6.21. 28 Jan 2004.
   http://enterprise.netscape.com/docs/directory/621/relnotes/ds621relnotes.html
   (10/04/2004)
6. "OpenLDAP 2.2 Administrator's Guide". The OpenLDAP Project. 25 Feb 2004.
   http://www.openldap.org/doc/admin22/
   (10/07/2004)
7. "Oracle Internet Directory Feature Overview". Oracle Internet Directory Service. 2002.

http://www.oracle.com/technology/products/oid/htdocs/oid_overview/oidfov9iASv2.html
(10/04/2004)
8. Rob Weltman and Tony Dahbura. LDAP Programming with Java. Addison Wesley, 2000.
9. M. Wahl, T. Howes and S. Kille. "Lightweight Directory Access Protocol (v3)". IETF RFC 2251. 1997.
http://www.ietf.org/rfc/rfc2251.txt
(09/26/2004)
10. "What is Active Directory?". Active Directory Overview. 30 Jun 1999.
http://www.microsoft.com/windows2000/server/evaluation/features/dirlist.asp
(10/04/2004)
11. Craig Zacker. "LDAP and the Future of Directory Services". WindowsITPro Magazine. 1997.
http://www.winnetmag.com/Article/ArticleID/251/251.html
(10/07/2004)

9 Acknowledgements

First of all, I would like to thank Dr. Dezhgosha for providing me with this excellent opportunity. Without his continuous encouragement and valuable guidance, this project and this paper would not have been possible.

I would also like to thank Dr. Keith Miller for his advice on concepts related to software engineering and software testing. His guidance about how to search, read and understand technical papers has helped me tremendously in this effort.

I thank the University Of Illinois and specifically, the Department of Computer Science, for providing all the resources, tools and help.

Last but not the least; I would like to thank my wife, my parents, my brother, my sister-in-law and my friends for their moral support and encouragement throughout this fantastic experience.