WERKSTATT UNTERNEHMENSSOFTWARE KARLSRUHE
(WUSKAR)

SYNCHRONISATION OF DIRECTORY SERVICES WITH THE EVENT PROPAGATION FRAMEWORK

Michael Manz
Christian Emig
Sebastian Abeck

Cooperation & Management (C&M)
Institut für Telematik
Universität Karlsruhe (TH)
wuskar@cm-tm.uka.de

Short Description

This case study introduces the most relevant directory service standard LDAP and the approach how to synchronize these directory services using the Event Propagation Framework (EPF) of iC Consult to establish a cooperate directory service as it is done at Daimler Chrysler.

Keywords

WUSKAR, Event Propagation Framework (EPF), Lightweight Directory Access Protocol (LDAP), Synchronization

Learning Goals

(1) Understanding the principle of LDAP directory services

(2) Understanding how LDAP works

(3) Understanding the Event Propagation Framework (EPF) and the underlying architecture

(4) Learn how to synchronize LDAP directory services with the Event Propagation Framework

Learning Goals 1: SYNCHRONISATION OF DIRECTORY SERVICES WITH EPF

(1) The goal is the comprehension of the structure and the function of an LDAP directory service, beginning with the understanding of directory services in general, and why LDAP is needed.

(2) Another aim is to advance the knowledge of LDAP directory services by learning the historical roots of LDAP, and how LDAP stores, processes and manages the information its directory supplies. Some basic information about the models making up LDAP and further knowledge of how to establish a LDAP server will be given.

(3) An impression of the Event Propagation Framework (EPF) is given by understanding the underlying architecture and the internal structure. Then information on how to configure and run the EPF is presented.

(4) The final goal of this case study is to learn and understand how to establish two LDAP servers and synchronize them using the EPF.

Table of Contents

Information 2: SYNCHRONISATION OF DIRECTORY SERVICES WITH EPF - Content Overview

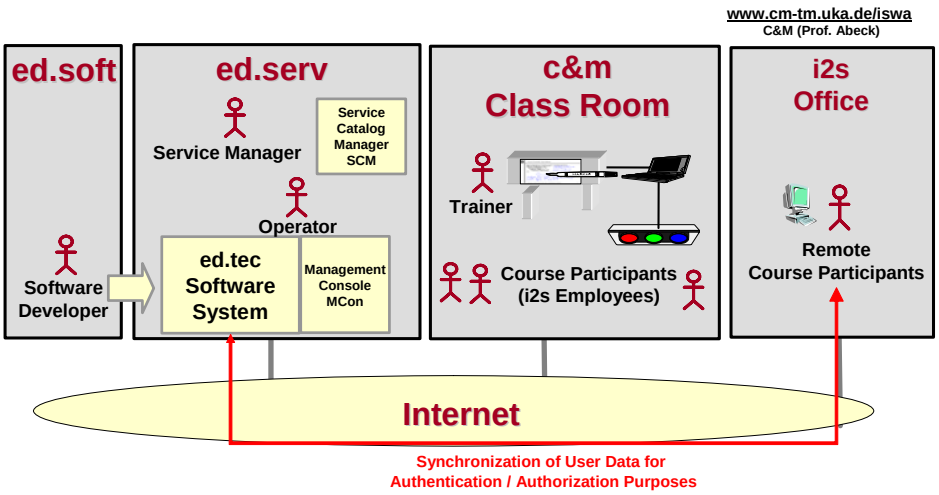
(1.1) A short introduction is given by placing the goal of this document in the standard ISWA scenario.

(2) This chapter explains the LDAP and its function within directory services.
(2.1) This chapter explains briefly the structure and the function of a directory service, and why LDAP is a good choice for accessing it.
(2.2) The knowledge of LDAP can be advanced by learning about its historical roots and its structure in detail.
(2.3) LDAP consists of four models, which are briefly described in this chapter.

(3) An impression of the Event Propagation Framework (EPF) can be developed by understanding the underlying architecture and a more detailed description of the EPF components Source and Destination.

(4) A description of how to do directory service synchronization with the Event Propagation Framework is presented. First two LDAP servers are installed, then it is explained how a simple EPF application is installed and configured. Finally it is described how to run this application performing synchronization with LDIF file.

1 INTRODUCTION TO THE SCENARIO



(1) One challenge in this scenario: i2s employees have to authorize to c&m

Information 3: INTRODUCTION TO THE SCENARIO – Scenario of IT-Based Training in the ISWA Lecture

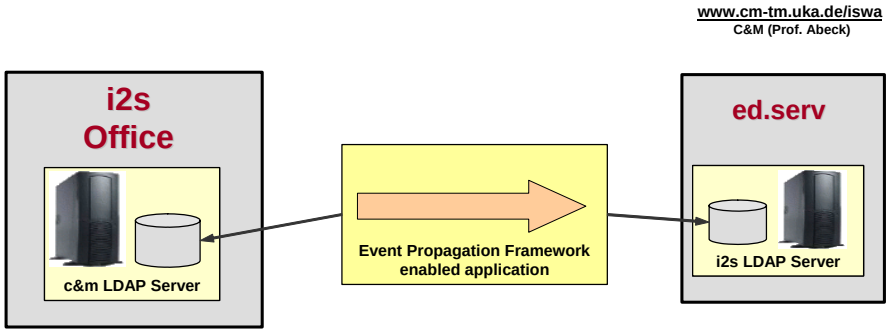
c&m (cooperation & more): Training company which offers IT-based training using internet / intranet

i2s: (intelligent internet solutions): Customer of c&m

ed.serv: IT service provider of c&m

ed.soft: Software company which has developed the ed.tec system (and related management solutions)

(1) i2s employees want to log into c&m's systems to have access to the course material. One of the challenges in this scenario is that i2s employees have to be authorized to c&m to access the courses material. In worst case, the user login information is copied manually from i2s to c&m. Besides the disadvantage of wasting human resources, no automatic updating of user information is offered using this method.



(1) An Efficient solution: LDAP driven directory service synchronization with EPF

Information 4: An Efficient Data Synchronization Solution

(1) An efficient solution is to use directory service synchronization with the EPF to obtain the required information from i2s.

2 LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL (LDAP)

2.1 A short Introduction to Directory Services

www.cm-tm.uka.de/iswa
C&M (Prof. Abeck)

(1) A directory provides hierarchical structured information

(2) A directory is a distributed directory with remote access

(3) The information is stored in the directory information base (DIB)

(4) An entry in the DIB is called an object class

(5) Directory user agents (DUA) enable access to the directory

(6) Directory services provide decentralized data storage and management

Information 5: LIGHTWEIGHT DIRECTORY ACCESS PROTOCOL - Overview of Directory Services

(1) A directory service provides hierarchical structured information and follows the X.500 standard using an object-oriented data model and a network protocol. X.500 was originally developed by a telephone company to be able to compile a standardized international electronic phone book.

(2) According to X.500, a directory is a distributed directory with remote access.

(3) The information that is stored in the directory is called the directory information base (DIB). The DIB is structured as a tree, which is called the directory information tree (DIT).

(4) Each entry in the DIB (or DIT) is called an object class, which consists of several attributes. Attributes are either must-attributes that each of the object class instances must have and may-attributes that can be defined for an instance, but could also be omitted when the object is created. Attributes are defined as stand-alone schema entities. They exist without being part of an object and have a fixed type perhaps with more than one value. Aliases are links to existing entries. They make entries available at more than one point in the DIT.

(5) Directory user agents provide operations for reading, comparing, searching, adding, deleting and modifying entities.

(6) Various directory servers have to communicate with each other to forward requests to the server where the requested information is stored. This means that sub-trees of the DIT can be hosted by different servers which are called directory system agents (DSA).

As mentioned a directory service organizes content in a directory server into a logical and accessible structure. It provides a single, consistent database in which information about the network and all network-based resources like users, servers, files, printers, shares, etc. can be stored.

(1) A directory service acts as a central authority

(2) Major differences between directory services and conventional databases are:
  (1) Information is read more often than it is written
  (2) There may exist entries or attributes that have values in different states
  (3) Transactions and rollback are not implemented in a directory
  (4) Data is organized in a strictly hierarchical manner

(3) Main criteria is to get a faster response time during searches
  (1) LDAP as a fast and simple way to access directory services

Information 6: A Closer View at Directory Services

(1) This authority can securely authenticate resources and manage identities and relationships between them. It defines the namespace for the network by mapping the names of network resources to their respective network addresses

(2.1) Information that is stored in a directory is generally read more often than it is written. The distributed nature of directories makes them inherently better at read requests as opposed to frequent updates.
(2.2) Because the directory is distributed, there may exist entries or attributes that have values in different states due to the delayed replication propagation.
(2.3) Also because of the distributed nature, these features, which are found in modern relational databases, are not implemented.

(3) The main criteria when designing a directory service is to get a faster response time during searches.

2.2 Accessing Directory Services with LDAP

(1) LDAP was developed at the University of Michigan

(2) LDAP is a protocol for accessing online directory services
  (1) Protocol elements are carried directly over IP (TCP / UDP)

(3) An entry in an LDAP directory is a set of attributes

(4) LDAP server store information in a tree-based structure that is also called directory information tree (DIT)

(5) LDIF file format is created to exchange data between LDAP servers

(6) LDAP directory servers are daemons that enable (remote) access to such an LDAP directory

Information 7: Overview of LDAP

(1) LDAP was designed to adapt X.500 to the modern Internet. X.500 is too complex to be quickly implemented, so LDAP was created to provide this a lightweight service.

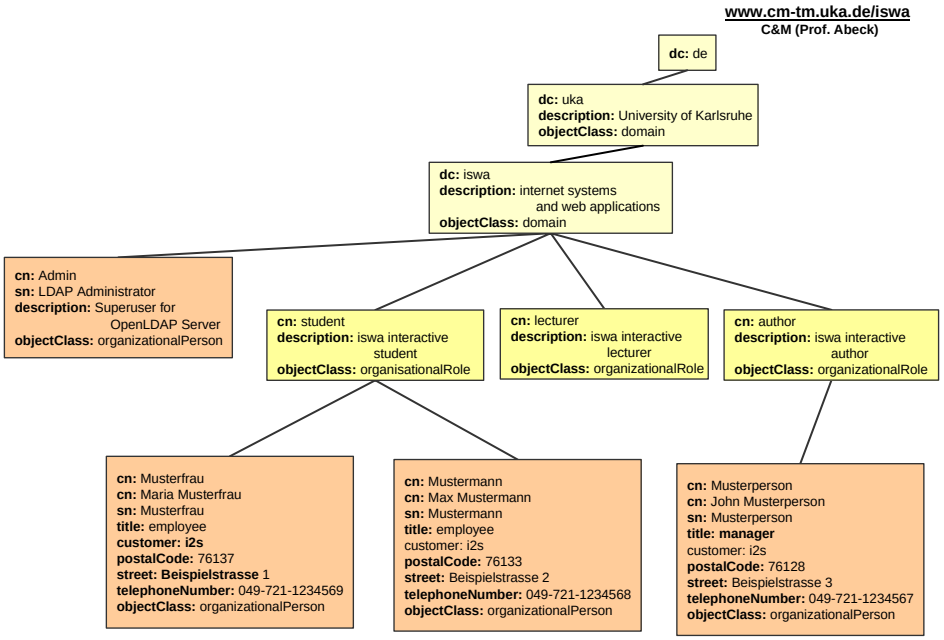
(2) LDAP is a protocol for accessing online directory services and to ease the use of X.500 directories. It is a relatively simple protocol for updating and searching directories running over TCP/IP. The term LDAP directory does not describe the type of the directory. It only implies that this directory is accessible using the LDAP protocol. Many protocol data elements are encoding as ordinary strings (e.g., Distinguished Names). A lightweight BER encoding is used to encode all protocol elements.

(3) Each of these attributes consists of a name and one or more values. Such an entry features a hierarchical structure that reflects political, geographical, and/or organizational boundaries.

(4) An example of such a DIT is shown in Information 8. The distinguished name (DN) in the DIT is, like the primary key in databases, an unambiguous identifier of each node. Starting at the root node, which is also called the base DN, every node extends the DN by another stage.

(5) LDAP Data Interchange Format (LDIF) is an ASCII file format used to exchange data and enable the synchronization of that data between LDAP servers. This will get very important later for the main learning goal of this case study.

(6) Popular LDAP servers are OpenLDAP Server, MS Active Directory, Novells eDirectory, Oracles Internet Directory and Sun (iPlanet) directory server.



Information 8: Example of a DIT

2.3 LDAP Models

LDAP is described by a combination of the following models:

www.cm-tm.uka.de/iswa
C&M (Prof. Abeck)

(1) Informational model: Describes the structure of information in a directory information tree (DIT)
  (1) Schema, Classes, Attributes, Syntax, Entries

(2) Naming model: Describes how information is organized and referenced
  (1) Distinguished Name, Relative Distinguished Name

(3) Functional model: Describes what can be done with the information
  (1) Authentication, Interrogation, Update

(4) Security model: Describes how information is protected in the directory information tree

Information 9: LDAP Models

(1) The informational model was derived from the ISO X.500 standard for creating enterprise-level directories. The informational model as defined in [IETF-LDAP1.0] and [IETF-LDAP3.0] describes entries and attributes

(1.1) A schema acts as a blue print or a template for the directory. The schema provides a listing of classes and attributes from which all entries are derived. A class is a category of objects that share a set of common characteristics. They are described by attributes, which are data items defined separately in the schema. The syntax of an attribute defines the storage representation,

byte ordering, and the matching rules for comparisons of property types. An entry is either a container or a leaf object of a specific structural class.

(2) The OSI directory model used Distinguished Name as the primary key for entries in the directory. The naming model is outlined briefly in [IETF-LDAP1.0] and [IETF-LDAP3.0]

(2.1) The Distinguished Name consists of a series of Relative Distinguished Names (RDN) and serves as a primary key for an object in the directory information tree.

(3.1) The functional model consists of nine operations in three areas. Authentication allows the client to prove his identity to the DSA. Interrogation provides a method for the client to interrogate the directory information tree. Finally, update defines a mechanism for the client to add or modify information in the directory information tree. The operations in this areas will not be discussed in this document, see [IETF-LDAP1.0] and [IETF-LDAP3.0] for details.

(4) The security model specifies how to access information in the directory in a secure manner. [IETF-LDAP3.0] states that Simple Authentication and Security Layer (SASL) mechanisms may be used with LDAP to provide association security services.

2.4 Exercises

www.cm-tm.uka.de/iswa
C&M (Prof. Abeck)

?

(1) Explain briefly the architecture of the EPF

(2) How do these components co-operate?

(3) What is the role of the source?

(4) Compare EPF's source and destination

Information 10: Exercises

3 EVENT PROPAGATION FRAMEWORK (EPF)

www.cm-tm.uka.de/iswa
C&M (Prof. Abeck)

(1) The EPF is a framework to propagate changes between directory services

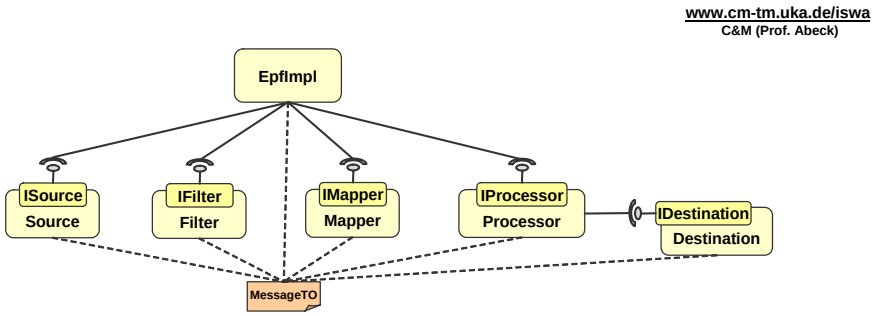
(2) The EPF provides necessary functionality to create a java application which afforts this topic

Information 11: EVENT PROPAGATION FRAMEWORK (EPF) - Introduction

(1) Currently this framework only works with LDAP driven directory services.

(2) In many cases such applications could be created fast and easily by just configuring the XML configuration file.

3.1 Underlying Architecture



(1) EpfImpl represents a Mediator

(2) MessageTO represents a Change Event in the Source Data Storage

(3) Source encapsulates the Access to the Source Storage

(4) Filter deletes unwanted or unnecessary Information

(5) Mapper maps Attributes and/or Values from Source Notation to Destination Notation

(6) Processor executes further Transformation to fulfill additional Business Needs

(7) Destination encapsulates the Access to the Target Storage

Information 12: The Synchronization Process

Each element has to implement the corresponding interface. Because the interface elements in Information 12 know nothing about one another; the special mediator element EpfImpl is needed as a mediator to coordinate the communication between these elements.

www.cm-tm.uka.de/iswa
C&M (Prof. Abeck)

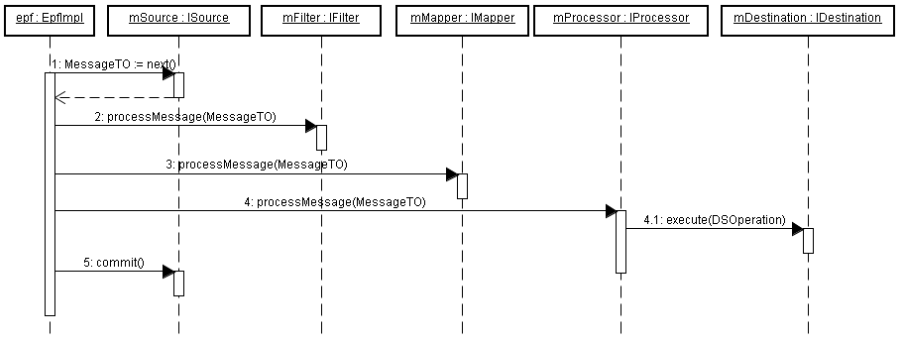
(1) External elements:
   (1) Controller: allows to manage an EPF based application provides over TCP
   (2) Monitor: allowing to monitor application health over SNMP

(2) XML based configuration file that contains setting information

Information 13: Further EPF elements

(1) EpfImpl also performs the communication with some EPF external elements, which were executed in separate threads like the Controller and the Monitor.

(2) An element that belongs to each EPF application is the configuration file. It is an XML based file that contains setting information about all elements controlling the assembling of the application during start-up.



Information 14: The Synchronization Flow

A successful synchronization flow from source to destination is shown in the sequence diagram in Information 14. The main task of the mediator EpfImpl is sending and receiving the globally used MessageTO object to each element.

(1) After declaring and implementing every class / element, a Message TO object is returned to the mediator EpfImpl using the method next(), which is defined as a global variable, that includes the changes, made in the according source. By calling the correlative method each component of EPF is invoked and is executing the changes on the MessageTO object:

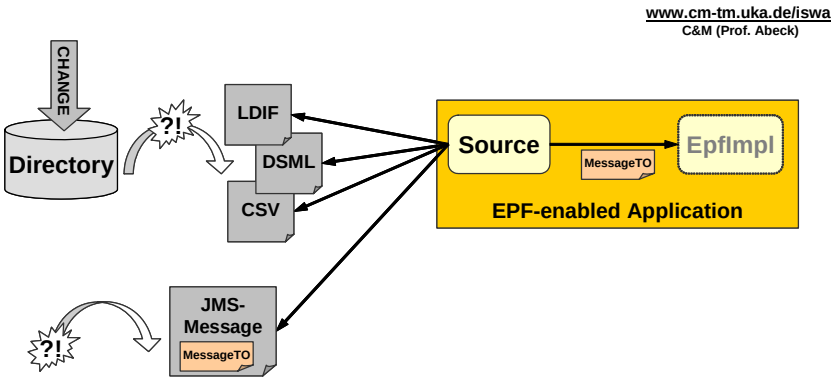
(2) The mFilter filters the preconfigured data from MessageTO that are needed and alters it in this way.

(3) mMapper maps the filtered information in the corresponding case.

(4) Finally mProcessor executes business needs on the further reached MessageTO.
(4.1) Here the altered MessageTO object is not sent upwards by EpfImpl to mDestination, but from mProcessor: the method execute conveys the changes to mDestionation. So the only way of communication between the mediator EpfImpl and mDestination is through mProcessor.

(5) If each element has executed its task the method commit() is called and the MessageTO object is deleted.

Source and Destination



(1) The Source component represents the access to the source storage
  (1) Serialization of many sources is managed by weights
  (2) FileBasedSourceExt, FileBasedSourceImpl, AbstractJMSSource, SourceDispatcher
  (3) Supported file formats: LDIF, DSML, CSV

Information 15: The Source

The source directory produces the change files. The Source component parses these files and encapsulates the data in the MessageTO object for further processing in the framework.

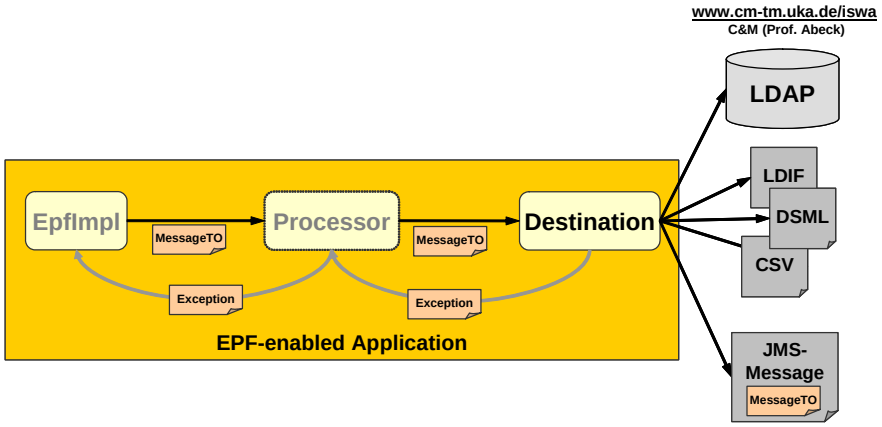
(1) The Source is responsible for getting information about changes in the source data storage and presents them in the message format.

(1.1) If many sources are used they have to be serialized. This is done by assigning a weight to every source in the source set. The application reads messages from this source depending on its weight w.

- w > 0: EPF will read exactly w changes from this source until switching to the next source.
- w <= 0: EPF will read all changes from this source until switching to the next source.

(1.2) FileBasedSourceExt generalises all file based sources like LDIF, CSV or DSMLv1 and DSMLv2. AbstractJMSSource is an interface to JMS queues. SourceDispatcher provides a way to get changes from different sources and set them in order. FileBasedSourceImpl is a deprecated version of FileBasedSourceExt and will not be discussed any further. Every source in SourceDispatcher is encapsulated in a SourceEntry. A SourceEntry can be weighted depending on its importance.

(1.3) DSML was designed to represent directory service information in XML syntax and provide an easy way to share respectively use personalized data across company and technology boundaries. CSV means coma-separate-value, a simple and common known file type.



(1) Destination component embodies the target storage
  (1) LDIFDestination, LDAPDestination, EmptyDestination

Information 16: The Destination

(1) The destination is responsible for updating information in the destination data storage.

(1.1) These classes allow executing the changes given by a global MessageTO object that describes the changes made on the source, on the respective destination. EmptyDestination is only a dummy for other storage possibilities. LDAPDestination points two modes of executions of changes: the read-only mode that provides bind, search or read, and the update mode, that provides add, delete, modify, rename, in log files. Using these, the destination can produce LDIF files which are useful in rolling back changes.

3.2 Exercises

www.cm-tm.uka.de/iswa
C&M (Prof. Abeck)

(1) Explain briefly the architecture of the EPF

(2) How do these components co-operate?

(3) What is the role of the source?

(4) Contrast source and destination

Exercises 17: Excersises

4 SYNCHRONIZING TWO LDAP SERVERS WITH EPF

First extract the enclosed archive to a folder of your choice. The path used to unzip this archive is referred to as $ZIP_ARCHIV further in this document.

4.1 Installing two OpenLDAP Servers

The OpenLDAP Server is available for Unix-based Systems. The rest of this subchapter deals with the installation and configuration of this server in a system with GNU Debian Linux.

After installing all necessary packages by executing the command

apt-get install slapd ldap-utils

in a shell, the configuration files have to be adjusted.

Without taking a closer look at these files, it should be mentioned that it is sufficient to edit the two files /etc/ldap/slapd.conf and /etc/ldap/ldap.conf. Preconfigured files can be found at $ZIP_ARCHIV/LDAP/conf/. Now the server can be started with the command

/etc/init.d/slapd start.

Of course the LDAP server does not contain any data. With the file $ZIP_ARCHIV/LDAP/conf/structure.ldif ceates an exemplary DIT can be created using the command

ldapadd -x -v -D "cn=Admin,dc=uka,dc=de" -W -f $ZIP_ARCHIV/LDAP/conf/structure.ldif

Searching in the LDAP directory can now be done by using ldapsearch:

ldapsearch -x cn=Admin.

4.2 Installing and Configuring the EPF

In this chapter a simple example is used to demonstrate the usage of EPF. In this example a LDIF file embodies the source as well as the destination storage.

4.2.1 The Structure of the EPF Distribution

The files and directories at $ZIP_ARCHIV/iCC-EPF/ have the following meanings:

- The source file are at $ZIP_ARCHIV/iCC-EPF/src and the complied files at $ZIP_ARCHIV/iCC-EPF/classes
- The folder $ZIP_ARCHIV/iCC-EPF/libs contains all necessary external JAR archives
- The start and stop scripts are included at $ZIP_ARCHIV/iCC-EPF/bin
- The folder $ZIP_ARCHIV/iCC-EPF/misc contains the configuration files
- All documentation is in the $ZIP_ARCHIV/iCC-EPF/docs folder
- $ZIP_ARCHIV/iCC-EPF/META-INF hosts the manifest file, which is needed when creating the JAR archive out of the EPF classes
- $ZIP_ARCHIV/iCC-EPF/.temp mainly contains the temporary files, which are generated during executing an EPF application
- The files $ZIP_ARCHIV/iCC-EPF/version.properties, $ZIP_ARCHIV/iCC-EPF/ant.properties and $ZIP_ARCHIV/iCC-EPF/build.xml are used when building the whole project using Apache Ant.

4.2.2 Configuring a first Application

To run an EPF-enabled application, at least the Java Runtime Environment (JRE) has to be installed. Because of several reasons it is very useful to also have Apache Ant installed, but this is not mandatory.

Using any IDE like Eclipse or Netbeans the following has to be done:

- Importing the folder $ZIP_ARCHIV/iCC-EPF/ to a new project
- Making the JAR archives at $ZIP_ARCHIV/iCC-EPF/libs available to the project
- Configuring in-/output paths

All necessary files are available at $ZIP_ARCHIV/iCC-EPF/. The configuration file is located at $ZIP_ARCHIV/iCC-EPF/misc/config.xml. Here the attributes /epf-config/sourceset/sourceentry/source/properties/entry[@key="data.path"] and /epf-config/destination/properties/entry[@key="output.path"] have to be adapted.

4.3 Running the EPF Application

When the configuration file is properly edited, the EPF application can be started by running the batch file $ZIP_ARCHIV/iCC-EPF/bin/start.bat. Now the application listens at the Source.
When a source LDIF file is placed at $ZIP_ARCHIV/iCC-EPF/.temp/ldif/ EPF will recognize this and write these changes to the destination file at $ZIP_ARCHIV/iCC-EPF/.temp/testout-*. After this little test is performed the EPF-enabled application has to be stopped by using $ZIP_ARCHIV/iCC-EPF/bin/stop.bat to get access to the created destination file. The content of this file should be similar to the content of the source file. The differences occur because of the filtering and mapping rules in the configuration file. All logging is also done in the folder $ZIP_ARCHIV/iCC-EPF/.temp/. If something went wrong during the execution, the log file may help to solve the occurring problems.

4.3.1 Creating a LDIF File

Because it is far from easy to create a LDIF file from a LDAP server, a simple LDIF Exporter found at $ZIP_ARCHIV/LDIFExporter was added.

(1) Limitation of LDIF file format: no possibility to export only the changed entries

Information 18: LDIF Export

In order to create a LDIF file from a LDAP server, the following steps must be taken:
Enter the host name of the LDAP Server, the login DN with password. Further define a search base and the search filter. Then, define the destination LDIF file. If the path is not absolute, the output will be created depending on the directory of the LDIF Exporter. Finally, press the "Export!" button. The LDIF Exporter produces a small LDIF file with the default values, because openldap.com operates a public test server.

(1) A problem of the LDIF file format is that it exports the whole content of the search base (apart from the filtered information). There is no possibility to request only the changed entries. This is not appropriate for professional solutions, so further solutions are needed. A discussion of such solutions would go far beyond the aims of this document.

TABLES

Abbreviations and Glossary

Abbreviation or Term	Full Name and/or Term Description
BER	*Basic Encoding Rules* Set of rules for encoding ASN.1 defined data into a particular representation for transmitting to another system.
Destination	A destination is a component in the EPF context that is responsible for update information in the destination data store. It has to implement, like each component of EPF, the respective interface.
DIT	Directory Information Tree The LDAP naming model defines how entries are identified and organized. Entries are organized in a tree-like structure called the Directory Information Tree. Entries are arranged within the DIT based on their distinguished name.
DN	Distinguished Name A Distinguished Name is a unique name that unambiguously identifies a single entry. DNs are made up of a sequence of relative distinguished names.
DSA	Directory Service Agent
DSML	Directory Service Markup Language The Directory Services Markup Language v1.0 (DSMLv1) provides a means for representing directory structural information as an XML document. DSMLv2 goes further, providing a method for expressing directory queries and updates (and the results of these operations) as XML documents [OASIS-DSML2.0].
EPF	Event Propagation Framework The EPF is a framework that provides necessary functionality to create java application that will synchronize content between data storages. The data synchronization process between two systems could be divided into 3 major parts: the data should be read from the source storage, somehow processed and written to the destination data storage. EPF has got the following components: Source, Filter, Mapper, Processor and Destination.
EpfImpl	EpfImpl is the mediator in EPF that connects the components like source, filter, mapper, processor with each other and is transferring the MessageTO object.
Filter	Filtering is the first stage (not mandatory) of change processing in EPF. It has four types of filtering changes, e.g. depending on the scope in the DIT , with possibility to customize two[iCC-EPF0.2].
LDAP	Lightweight Directory Access Protocol LDAP is designed to provide access to directories supporting the X.500 models, while not incurring the resource requirements of the X.500 Directory

	Access Protocol (DAP). This protocol is specifically targeted at management applications and browser applications that provide read/write interactive access to directories. When used with a directory supporting the X.500 protocols, it is intended to be a complement to the X.500 DAP [IETF-RFC2251].
LDIF	LDAP Data Interchange Format The file format, known as LDIF, for LDAP Data Interchange Format, is typically used to import and export directory information between LDAP-based directory servers, or to describe a set of changes which are to be applied to a directory [IETF-RFC2849].
Mapper	Mapper is a component in EPF responsible for mapping attributes from Source to Destination notation [iCC-EPF0.2].
MessageTO	MessageTO is a class in EPF used to encapsulate a change event in a source directory to propagate it to a destination directory [iCC-EPF0.2].
OSI	Open System Interconnection
Processor	This part is responsible for the EPF application specific business logic. It can also be empty. For integration the custom processor should implement a simple interface [iCC-EPF0.2].
RDN	Relative Distinguished Name A Relative Distinguished Name is part of a Distinguished Name and corresponds to a branch in the DIT leading from the root of the DIT to the directory entry. Each RDN is derived from the attributes of the directory entry. In the simple and common case, an RDN has the form <attribute name> = <value>. A DN is composed of a sequence of RDNs separated by commas [TE+04].
Source	Source is a component in EPF responsible for getting information about changes in a directory. Each Source is a class, which has to implement the ISource interface [iCC-EPF0.2].
WUSKAR	Werkstatt UnternehmensSoftware KARlsruhe
XML	eXtensible Markup Language XML is a standardized markup language by W3C to describe the structure of documents.

Index

Information and Exercise Slides

References

[ICC-EPF-UM0.9] iC Consult: Event Propagation Framework (EPF) - Usage Manual 0.9, iC Consult, 2005.

C&M: Application Integration (iC Consult)

[IETF-LDIF1.0] The Internet Engineering Task Force: Request for Comments 2849 - The LDAP Data Interchange Format (LDIF) - Technical Specification, http://www.faqs.org/rfcs/rfc2849.html, June 2000.

[IETF-LDAP3.0] The Internet Engineering Task Force: Request for Comments 2251 - Lightweight Directory Access Protocol (v3), http://www.faqs.org/rfcs/rfc2251.html, December 1997.

[IETF-LDIF1.0] The Internet Engineering Task Force: Request for Comments 2849 - The LDAP Data Interchange Format (LDIF) - Technical Specification, http://www.faqs.org/rfcs/rfc2849.html, June 2000.

[OASIS-DSML2.0] Organization for the Advancement of Structured Information Standards (OASIS): Directory Service Markup Language v2.0, http://www.oasis-open.org/committees/dsml/docs/DSMLv2.doc, December 2001.

[OPEN-LDAP] Official Homepage of OpenLDAP http://www.openldap.org/

[Ta02] Andrew S. Tanenbaum: Distributed Operating Systems, Prentice-Hall, 2002.

[MS2000] White Paper, Microsoft 2000