LDAP replication at CERN

Petr Matousek
CERN - IT Division
e-mail: Petr.Matousek@cern.ch

29.7.1998

Abstract

This paper is a technical report or a cook book from CERN describing our experiences and results with LDAP replication. Replication is one technics how to implement shared directory over LDAP protocol. Because of off-line update of our LDAP database we could not use standard replication daemon. However, we managed to modify parts of replication process in such way that our database is daily replicated. Another successfully solved issue is a translation of CERN local LDAP database (ldif format) to the shared format recommended for HEP sites.

1 Introduction

Requests from physicists community for a contacts to their collegues of collaborating institutes makes us to look for a solution how to connect local databases (directories) from different sites and build a global directory to be shared. The global directory should be transparent to the user and should contain up-to-date data. The latter requires daily update of database.

We decided to focus on the LDAP directory which becomes more and more popular and it is supported by major software producers. LDAP protocol is defined as a RFC standard and supports data distribution. It is based on X.500 standard and improves its features (simple implementation, security issues, access to WWW etc). Last version of LDAP protocol is described in RFC 2251 ([1]).

1.1 Problem specification

Before we choose any solution we should learn what users require from Directory service. Following list shows summary of major points:

User requirements

- Simple access - access from mail client, web browser etc.
- Up-to-date information
- Short responce time
- Transparency - no special skills or configuration needed

Few other points, more system related, are demanded by system administrators.

System administrator requirements

- Directory compliant with other products
- Simple data management
- Security - access control

Does LDAP directory cover all these requirements ? It depends on the server implementation, of course. Some of these points are recommended in LDAP specification, a part of them is product-dependent. Next chapter tries to answer these questions.

1.2 LDAP Directory

LDAP Directory is a client/server service based upon the LDAP protocol. LDAP protocol specifies communication between client and server. It defines server front-end only. Server itself can be implemented without constraints, there is not any recommended format of LDAP database.

An LDAP database is built on a Directory Information Tree (DIT) corresponding to the X.500 data model (see [2]). Directory information is stored in a *data store*.

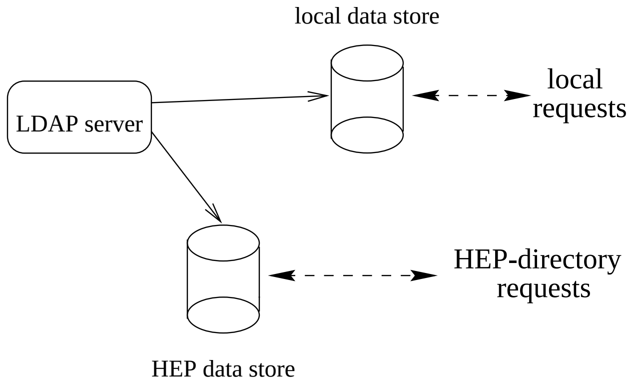

Figure 1: HEP site data stores

A server can contain more than one data store. Figure 1 shows a design of a site running an LDAP server. The server maintains both the local data store (updated from the main local database) and the global data store (a shared database). In this model we envisage independent sites with their local databases an extract of which is a part of a shared directory tree.

LDAP data (e-mail address, phone numbers etc.) amongst working sites of a large organization can be shared via the LDAP protocol in two ways:

- **Data replication**
  The replication process requires an additional server to check changes in the original data store and propagate them to other sites. We can specify the frequency and time of updates (when network load is low). The replication server connects via LDAP to a slave LDAP server and updates slave's database.

  Data replication is recommended only if the master LDAP database is rather stable and only a few entries are updated at a time. It is not a good idea to transmit the entire database every day over network.

- **Referrals**
  Referral is a pointer to another LDAP server specified in the local LDAP server configuration. When a server receives a request for information, it checks whether it can respond to the request. If it cannot, it checks the refferal and returns the details of

an alternate directory server to the client. The client can then send the request to the other directory server.

Using referrals we can avoid high-volume database replication. However, today not all LDAP clients can resend requests to other servers if a refferal is returned.

Because of immaturity of LDAP clients to process correctly referrals we decided to implement the global HEP directory based on replication rather then on referrals. We agree the replication is not the best solution but only one that can be used at present.

2 How replication works ?

Information from a master naming context is propagated to a replica (slave server) by the replication daemon.

A master server maintains a replication log. Each time the master database is updated, the transaction is recorded in the replication log. When replication daemon next runs, it reads the replication log and sends the change to the server that holds the replica naming context. The slave server handles update requests from replication daemon on master site in the same way that it handles all requests, using the information supplied in the bind request to set the access level granted to replication daemon. To guarantee that all replication updates are completed, replication daemon must bind with the DN defined when the replica naming context was configured. If a different DN is used, write access for all entries may not be granted.

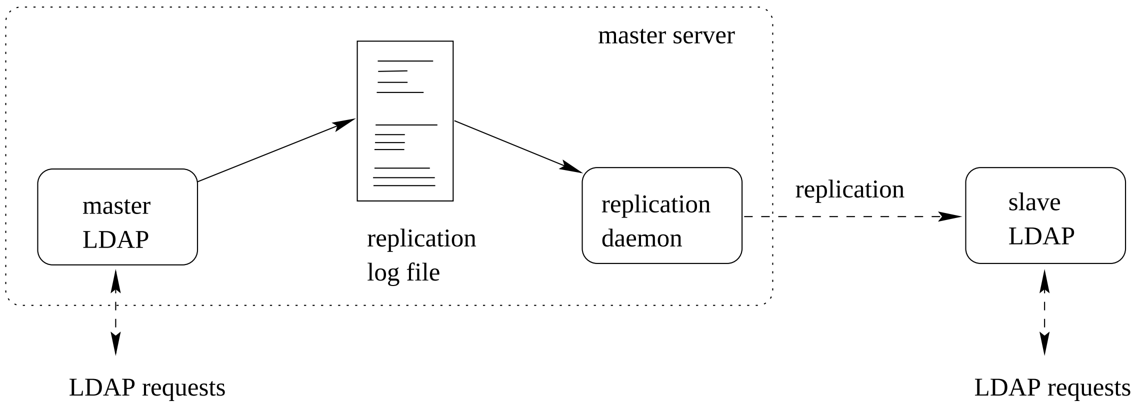

Figure 2: Replication schema

Any naming context held in the data store, including replica naming contexts, can be replicated to another server.

We can configure following options of replication process:

- What information is to be replicated - based on
  - DN of subtree
  - an individual entry (object)

- a filter

- What attributes are to be replicated
- The names of servers which hold replicas
- Synchronization schedule
  - Immediate - modification are made in the same time when master database is changed
  - Delayed - modifications are logged and replication server invoked according to a schedule (like cronjobs)
  - One-shot - modifications are logged and replications are propagated when replication daemon is invoked by hand

3 Shared directory vs. Local directory

It is a nice idea to share a directory but we shall realize one important thing: **Shared data must have the same format**. This can limit our local directory requirements. We want to provide local site-related data for local users only and an extract of local database shall be put to a shared directory.

We can do that in two ways:

- **One naming context**
  In this case an LDAP server holds only one naming context (shared one). Local data are in a subtree and contain more attributes. Access rules configuration defines visibility of these attributes for non-local clients - see Figure 3.

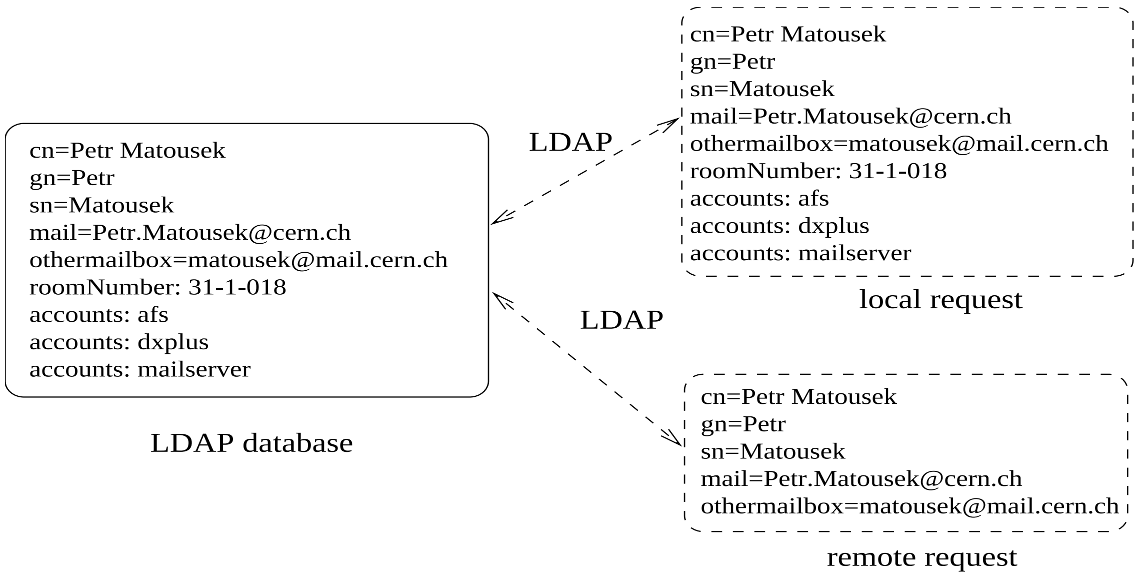

Figure 3: Shared directory with one naming context

  - Replication daemon shall be used
  - Database cannot be updated off-line (problem how to add shared data)

- **Two naming contexts (two data stores)**
  One data store holds a local data (with local attributes) and the second one a shared data (shared attributes). Extract of the local data is translated to be compliant with shared attributes definition and stored in the second data store. Configuration of local data store (attributes, DIT) can be different of the shared data store - see Figure 4.
  - Replication daemon can be used
  - Local database can be updated off-line ⇒ replication process have to be modified (manual creation of replication log).

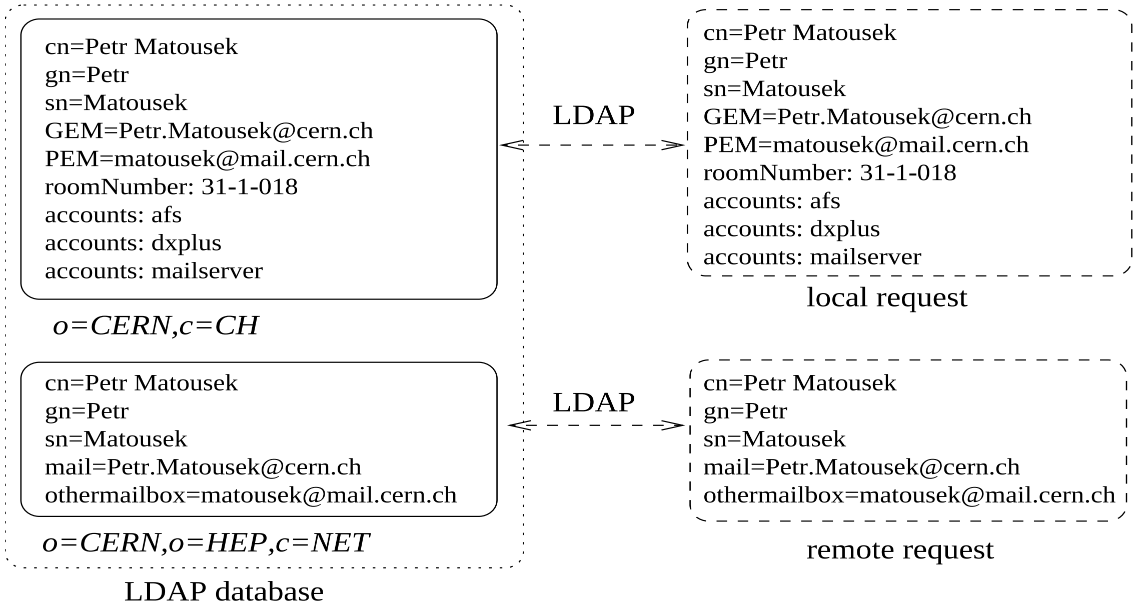

Figure 4: Shared directory with two naming contexts

3.1 Off-line update and replication

As we mentioned in chapter 2 the replication process uses update information from a log file maintained by the master server. Every replace/modify/delete operation is recorded in the replog file and then propagated (using LDAP protocol) to slave server.

If we update local LDAP directory off-line (creating db files in database internal format) we **cannot record changes using replication daemon**. Changes are done directly by generating new db index files, not by LDAP update requests.

Only way how to create a replication log is a comparison of the last version and the new version of LDAP database (in LDIF format) and saving differences to a log file (in LDIF format). After that we can start the replication daemon in usual way.

For this task we use a set of perl scripts - see Figure 5.

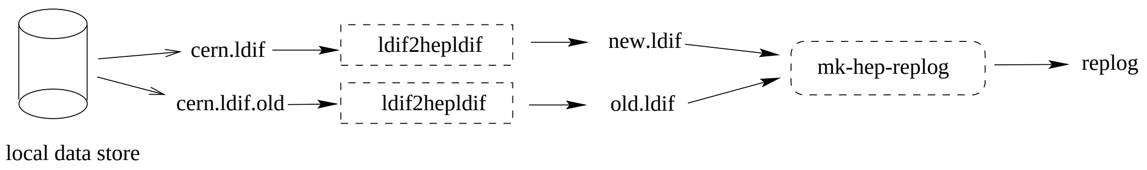

Figure 5: Off-line replication

Building of a replication log file has following phases:

1. **Transformation of a local database in LDIF format to HEP-LDIF format**[1]
   Names of local attributes are changed to HEP-format (e.g. *GEM* to *mail*, *PEM* to *mailbox* etc.). The operation is applied to the current version of database as well as to the last version (to find what was changed we need two last copies to be compared). Some local attributes (*account*, *office*, *room numbers* etc.) are abandoned.

2. **Creation of a replication log file**
   Replication log file records all differences between old and new version of LDAP database. Obsolete entries are deleted, new ones added or updated. Format of replication file is specified in [3] and [4].

4 CERN - State of arts

At CERN we have two LDAP servers - one holding CERN local directory and one test machine containg HEP directory - see Figure 6.

Local CERN Directory

- Site-dependent attributes (*PEM*, *GEM*, *accounts*, etc.)
- Everyday off-line update (input taken from Oracle database)
- Access rules based on domain addresses - special access granted to some HEP sites.

HEP Directory

- Still under evaluation
- Data replicated everyday from CERN local directory
- Server configuration (attributes, object classes) follows HEP-profile (see chapter 5.2).

4.1 Replication process

Replication requires co-operation of both sites - master server (holding original copy of database) and slave server (replica) - a data store which is database replicated to.

To establish replication we need first to configure both sides, synchronize database contents and then start replication process.

[1]described in chapter HEP-profile, page 11

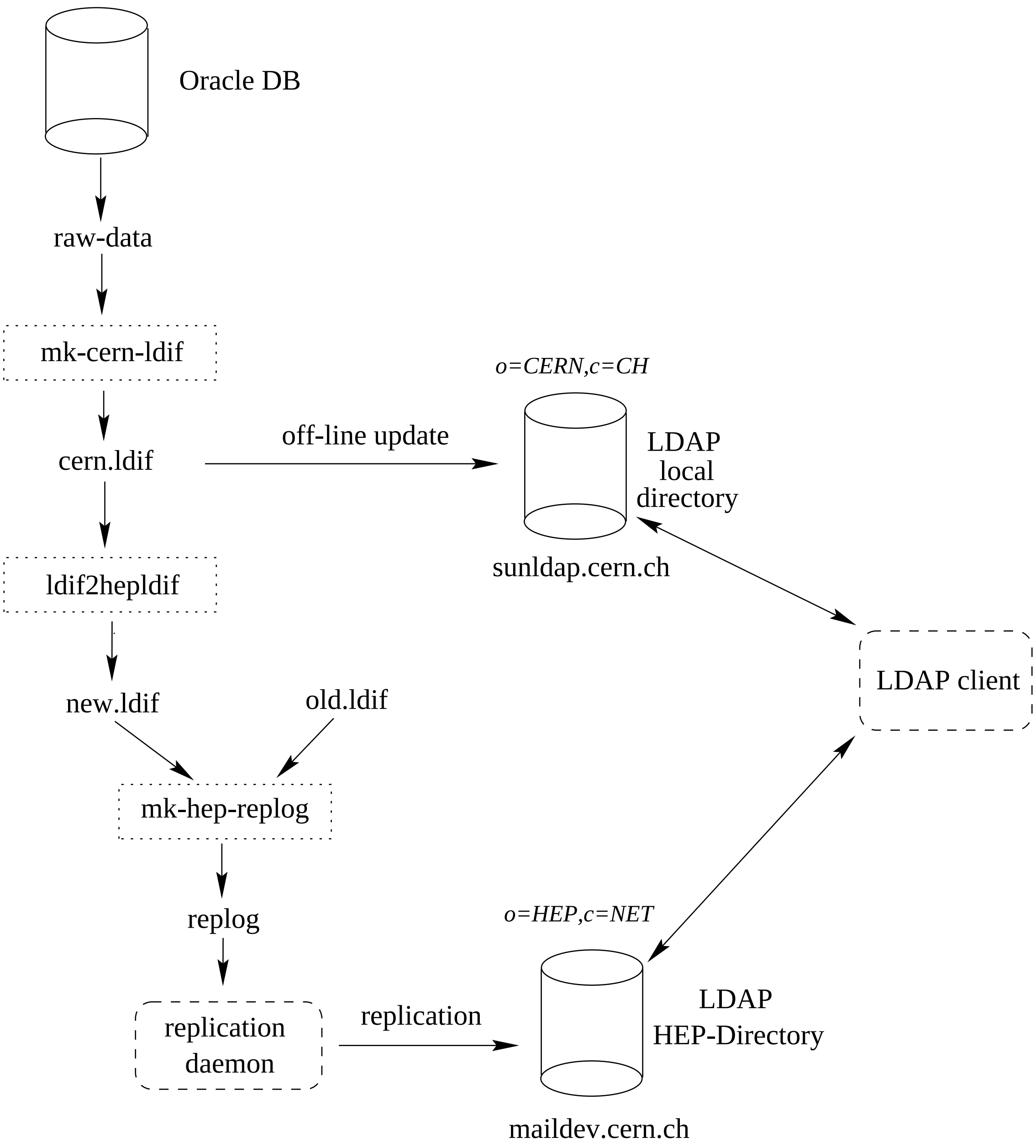

Figure 6: LDAP directories at CERN

4.1.1 Configuration

This part is more product-related. At CERN we use Sun Directory Server v1.0 which has similar configuration files as the SDS v3.0, or the LDAP server of University of Michigen.

- *Master server* - configuration file *slapd.conf*

```
...
replogfile /tmp/ldap/replog.log         # replication log file
...
replica host=maildev.cern.ch:389        # slave server
binddn="cn=replica,o=HEP,c=NET"         # DN performing updates
bindmethod=simple credentials=password  # password for bind
...
```

- *Slave server* - configuration file *slapd.conf*

```
...
access to dn=".*o=CERN,o=HEP,c=NET"     # write permission for replica
by dn="cn=replica,o=HEP,c=NET" write
...

slave
replicadn="cn=replica,o=HEP,c=NET"      # replica DN
referral="ldap://sunldap.cern.ch"       # master server
subtree="o=CERN,o=HEP,c=NET"            # replicated subtree
...
```

4.1.2 Synchronization

Replication is a process which propagates the update of the master context only. Before we start to run replication regularly we need to build up a slave context. When this is done we can propagate future changes over LDAP protocol.

We can prepare a slave database in two ways:

- *Off-line* - A copy of a master data (a dumb of the master db in LDIF format) is transferred (e.g. via ftp) to a slave machine and database created off-line by special program (*ldif2ldbm* for Sun Directory Server).

- *On-line* - Entire master database is propagated to the slave (replication deamon is invoked) first. Then changes are replicated when the master db is updated. This is done via LDAP protocol and it consumes much resources (network connection) and spend a lot of time. Not every LDAP server supports this solution.

We strongly recommend to use the first way.

Some LDAP servers (slaves) are able to pull out a data from master context. This way of replication was never tested and it is not discussed here.

4.1.3 Before running replica

Be aware that only users data are replicated. It is expected a replica server **does already contain** a root entry, HEP-site entries and an entry allowing updates (replica DN). These few entries we have to put after we start LDAP server with configuration mentioned above, before starting replication. CERN entries look as follows:

```
dn: o=HEP,c=NET                # root entry
objectClass: organization

dn: cn=replica,o=HEP,c=NET    # replica DN entry
sn: in2p3
objectclass: person
objectclass: top
userpassword: password
cn: replica

dn: o=CERN,o=HEP,c=NET        # HEP-organization entry
objectclass: HEP-organization
objectclass: top
l: Geneva, Switzerland, Europe
description: European Laboratory for Particle Physics
labeledurl: http://www.cern.ch
postaladdress:: Q0VSTgpGLTAxNjMxIENFUk4gQ2VkZXgB
postaladdress:: Q0VSTgpDSC0xMjExIEdlbmV2ZSAyMwpTdWlzc2U=
o: CERN
telephoneNumber: +41-22-76 xxxxx

dn: ou=People,o=CERN,o=HEP,c=NET # subtree of the CERN tree
objectclass: organizationalunit
objectclass: top
ou: People
```

4.1.4 Replication step by step

Summary of previous paragraphs:

1. **Configuration on both master and slave sites** - definition of replica server, DN to be allowed to update database, attributes and object classes of replicated data have to be defined on a slave server

2. **Slave database - initialization** - a copy of the master database is created

3. **Slave server starts running**

4. **"Top" entries are inserted into the slave database** (*ldapadd*) - root entry, organization entry (top of subtree), replica entry

5. **Replication is started** - by running replication daemon (*slurpd*).

4.2 Statistics

This section gives a brief information how often is CERN database updated and how many entries are affected by this changes.

day	updates	add	del	mod	modrdn
Mon	145	13	45	170	0
Tue	125	36	20	122	0
Wed	5881	12	30	7258	3
Thu	259	17	35	254	2
Fri	112	8	50	99	0
Sat	16	2	1	16	0
Sun	0	0	0	0	0
Avg	**1080**	**14**	**27**	**1309**	**1**

Figure 7: Replication traffic

A Figure 7 shows how many entries are updated every day. Data in the table are average day of the week which were collected in 18 days we are running the replication regularly. The table contains number of updated entries (*updates*) and number of LDAP operations (*add,del,mod,modrdn*). Only 14 entries are added every day and 27 are deleted. The most frequent operation is a modification. Modified are in particular attributes *home institute* - 87%,*phone number* - 5%, *division* - 4% and *mail* - 2%.

Time of transmission and amount of a transmitted data is fairly reasonable. For instance a replication of 50 entries (10kB) took about 15 seconds.

5 Replication among HEP-sites

A main goal of our effort is to build a global directory to be shared amongst HEP sites to enable every scientist to find his colleague's e-mail address by a single request without knowing his working place.

A directory running over LDAP protocol was found the most promising solution for this task. At CERN we made a pilot test with replication amongst two servers - results are described in previous chapters. We are going to test a replication between two remote sites - CERN in Geneve and IN2P3 in Lyon.

5.1 HEP-recommendation

If we want to share a data we have to specify how to share them and what kind of data to share. A strict boundaries have to be defined for those who are involved in a replication. After HEPiX meeting in Lyon (April 1998) and discussion with Helene Jamet from IN2P3 (reported at [5]) a following recommendation were issued. This is proposal to HEP community and it is considered as a base for further discussion. Any suggestion and improvements are warmly welcomed.

1. Each site SHOULD contain a copy of the global directory (to answer local requests) or to redirect all request to the nearest LDAP server (permission of that site is desirable).

2. A site MAY hold more naming contexts.

3. Each site MUST produce a replication log having entries within the naming context o=HEP, c=NET (e.g. o=CERN, o=HEP, c=NET).

4. We do not specify how the local database is built or what kind of LDAP server is used. However, LDAP server supporting protocol version 3 is strongly recommended.
5. Shared data (usually a subset of the local data set) MUST follow the **HEP profile schema** as described in Section 5.2.
6. Shared data SHALL be filled in the English language (especially a field *descriptions*).A format of postal address and phone numbers is specified by international telecommunication standards.
7. Each site is responsible for data it replicates - especially for validity and correct format.

5.2 HEP-profile

This section proposes a standard of HEP directory tree and recommended object classes. It updates recommendation described in [5]. Examples of entries can be found in Appendix A.

5.2.1 Class HEP-person

A person recommended entry shall contain the following attributes:

```
objectclass: HEP-person
subclass of: top
attributes:
   a)mandatory:          # MUST be filled
     - commonName (cn)
     - surname (sn)
     - givenName
   b)optional:           # MAY be filled
     - telephoneNumber          # multi-values attribute (international format)
     - organizationalUnitName (ou) # division, section etc
     - mail                     # preffered e-mail address
     - otherMailbox             # multi-values attribute
     - labeledURI               # e.g. URL of home page
     - businessCategory         # means experiment (e.g.Delphi)
     - organizationName         # home institute
```

5.2.2 Class HEP-organization

An organization recommended entry shall contain following attributes:

```
objectclass: HEP-organization
subclass of: top
attributes:
   a)mandatory:          # MUST be filled
     - organizationName (o)
   b)optional:           # MAY be filled
     - description
     - localityName (l)
     - postalAddress     # multi-value attribute if more sites
```

```
    - fax
    - telephoneNumber
    - labeledURI      # information about organisation (e.g. URL)
```

5.2.3 HEP-organizationalUnit

An organizationalUnit recommended entry shall contain following attributes:

```
objectclass: HEP-organizationalUnit
subclass of: top
attributes:
   a)mandatory:         # MUST be filled
     - organizationUnitName
   b)optional:          # MAY be filled
     - description
     - fax              # secretariat fax number
     - telephoneNumber  # secretariat phone number
     - labeledURI       # URL of HEP-ou
```

5.2.4 Class HEP-referral

A referral recommended entry shall contain following attributes:

```
objectclass: HEP-referral
subclass of: ref
attributes:
   a)mandatory:         # MUST be filled
     - ref
   b)optional:          # MAY be filled
     - description
```

5.3 HEP directory tree

A proposal of Directory tree was made as depicted in Figure 8.

Root context is *o=HEP,c=NET*. Its subtrees represent directories of each HEP site. A format of distinguished name of entries in HEP site subtrees is not defined. Every site can present its own structure based on geographical or organizational hierarchy.

A distinguished name of a person entry **SHALL be composed of the commen name of a person and a unique ID** (e.g. *dn="cn=Petr Matousek id=007, ou=People, o=CERN, o=HEP, c=NET"*).

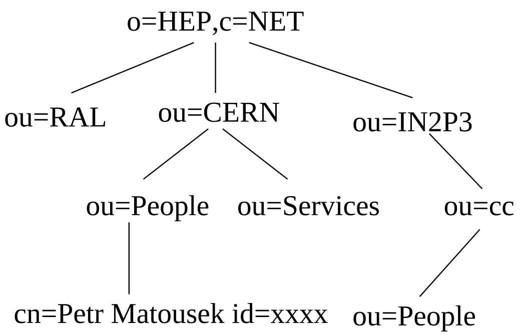

Figure 8: HEP directory tree

5.4 Replication topology

- *How shall we distribute data ?*

To replicate data from every single site to other sites is not very effective. It requires to establish $n\ x\ n$ connection, where n is number of nodes sharing global directory. More effective is to define path where data flows from one site to another. In selection of that path many parameters are to be considered to make replication fast and reliable. We can use topology called in graph theory **a spanning tree**.

5.4.1 Replication tree

In Figure 9 a possible topology is depicted using *a spanning tree*. Highlighted lines form *a minimal spanning tree* used for replication. Edges of the tree represent bi-directional connections for data propagation.

The spanning tree shows the path by which replicated data is distributed. How to find a minimal spanning tree ?

- First we create a complete graph containing every node.
- Then we evaluate every edge using cost function. The function depends on many factores - network load, distance, connection bandwith and througput etc.

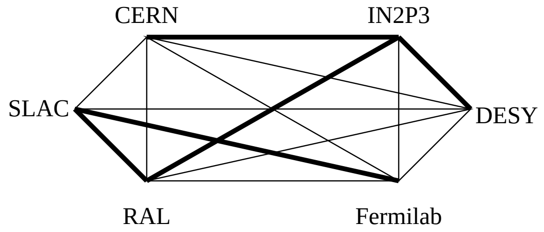

Figure 9: Replication path

- After evaluation we search for *the minimal spanning tree* and set a direction for distribution and a subset of data are to be distributed. For instance, CERN (ending node) replicates only its data to IN2P3. It receives replicated data from the remaining sites.

Figure 10 shows a detail of data flow between nodes IN2P3 and RAL.

Both server hold entire global context *o=HEP, c=NET*. But only parts of the context are replicated. IN2P3 sends changes from CERN, DESY and itself. It replicates subtree *o=CERN, o=HEP, c=NET*, *o=IN2P3, o=HEP, c=NET* and *o=DESY, o=HEP, c=NET*.

In opposite direction it receives changes of following subtrees *o=SLAC*, *o=RAL* and *o=Fermilab*.

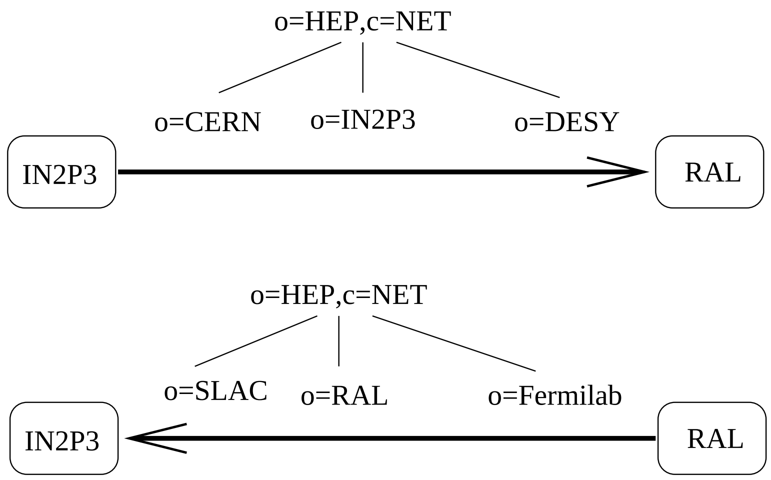

Figure 10: IN2P3 - RAL traffic - example

LDAP server usually supports replication of selected subtree, so no more administrator's effort is needed. Just connection to a global directory network requires changes in server configuration.

5.4.2 Maintenance of the tree

Maintenance of the topology should not be complicated.

- **New node added**
  - Every site creates "top" entry of the new site (e.g. *o=FNAL,o=HEP,c=NET* and synchronize data itself as a slave (see chapter 4.1.2 and 4.1.3).
  - Neighbour node will receive a data from the new node and replicates them to other sites.
  - Neighbour node add the new node to a set of nodes it replicates data to.
- **Existing node removed**
  - Every site discards a subtree belonging to a node to be removed (obsolete data are not useful and can be misleading).
  - Neighbour node removes the node from a set of nodes it replaicates data to.
- **Existing node is temporary unavailable**
  - Every site keeps a subtree of this node. No updates are performed.
  - Neighbours of the node "skip" the node in replication strategy and will communication with each other according to a minimal spanning tree structure.

6 Conclusion

References

[1] M.Wahl, T.Howes, S.Kille: Lightweight Directory Access Protocol (v3), RFC 2251, December 1997.

[2] ITU-T Rec. X.501 - The Directory: Models, 1993.

[3] The SLAPD and SLURPD Administrator's Guide - release 3.3, University of Michigan, April 1996.

[4] SDS 3.1 Administration Guide (Beta draft), March 1998

[5] H.Jamet, P.Matousek: The Global Directory among HEP sites, meeting report, June 1998.

Appendix A - examples of HEP entries

- **HEP-person**
  dn="cn=Petr Matousek, id=007, ou=IT,ou=People, o=CERN, o=HEP, c=NET";
  cn: Petr Matousek
  sn: Matousek
  givenName: Petr
  telephoneNumber: +41-22-767 9801
  ou: IT/DIS
  mail: Petr.Matousek@cern.ch
  othermailbox: matousek@mail4.cern.ch
  othermailbox: matousek@maildev.cern.ch
  othermailbox: matousp@dcse.fee.vutbr.cz
  labeledURI: http://www.cern.ch/~matousek
  businessCategory:
  o: Technical University of Brno, Czech republic

- **HEP-organization**
  dn="ou=CERN, o=HEP, c=NET";
  o: CERN
  description: European Laboratory for Particle Physics
  l: Geneva, Switzerland, Europe
  postalAddress: CERN, CH-1211 Geneve 23, Suisse
  postalAddress: CERN, F-01631 CERN Cedex
  telephoneNumber: +41-22-76 761 11
  fax: +41-22-76 765 55
  labeledURL: http://www.cern.ch

- **HEP-organizationalUnit**
  dn="ou=IT, ou=CERN, o=HEP, c=NET";
  ou: IT
  description: Information Technology Division
  telephoneNumber: +41-22-767 5012
  telephoneNumber: +41-22-767 2391
  telephoneNumber: +41-22-767 2393
  fax: +41-22-767 7155
  labeledURI: http://wwwinfo.cern.ch

- **HEP-referral**
  dn="ref=ldap://mail2.desy.de:389/o=desy,c=de";
  ref: ldap://mail2.desy.de:389/o=desy,c=de;
  description: referral to DESY laboratory in Germany