August, 2006

Inventions on Data Searching in LDAP - A TRIZ based analysis

Umakant Mishra
Inventions on
Data Searching in LDAP
-A TRIZ based analysis

By- Umakant Mishra, Bangalore, India
http://umakantm.blogspot.in

Contents

Contents ........................................................................................................................................1
1. Introduction ......................................................................................................................................2
2. Study on data searching in LDAP .................................................................................................2
  2.1 Objectives of the study .............................................................................................................2
  2.2 Major areas of Invention ...........................................................................................................3
  2.3 Patents analyzed for the study ..................................................................................................3
3. Introduction to the Searching Mechanism in LDAP .................................................................4
4. Major Concerns in LDAP data searching ......................................................................................5
5. IFR for LDAP data searching .......................................................................................................5
6. Inventions on data Searching in LDAP ........................................................................................6
  6.1 Generating SQL query statements for LDAP Search Filter ....................................................6
  6.2 Reverse indexing LDAP database for faster searching ............................................................7
  6.3 LDAP directory server cache mechanism ...............................................................................8
  6.4 Implementing SQL in LDAP queries .....................................................................................10
  6.5 Multi-level directory searches in LDAP ...............................................................................11
  6.6 Scalable event notification in LDAP systems .........................................................................11
  6.7 Method of caching LDAP queries ..........................................................................................13
  6.8 Authentication referral search for LDAP ...............................................................................14
7. Other related inventions ...............................................................................................................15
  7.1 Data mapping in RDBMS for searching LDAP data ...............................................................15
  7.2 Invention-13: Automated distinguished name lookup .............................................................15
  7.3 LDAP-based distributed cache technology for XML ...............................................................15
  7.4 Searching profile attributes based on other target profile attributes and associated profiles .................................................................................................................16
8. Summary and Findings of the study ...........................................................................................16
  8.1 Distribution of patents .............................................................................................................16
  8.2 Hot areas in LDAP data searching ..........................................................................................16
  8.3 Trends of evolution in LDAP storage ......................................................................................17
  8.4 Predicting future inventions on LDAP searching .....................................................................17
Reference to patents: ......................................................................................................................18
Other references: ..........................................................................................................................19
1. Introduction

Lightweight Directory Access Protocol (LDAP) is an IETF open standard to provide directory services in the network. LDAP was initially developed at the University of Michigan with an objective to include most of the features of X.500, while eliminating the burdens and difficulties of the same.

The current Version of LDAP is LDAP V.3 released in December 1997 (RFC 2251). Other specifications of LDAP can be found in different RFCs on the IETF (Internet Engineering Task Force) website.

With the growing use of Internet, LDAP is becoming more and more popular to provide directory services to a wide range of applications. This led to patenting several inventions relating to LDAP operation and application. This study on LDAP data storage is a part of the main study on LDAP based on 60 selected patents on LDAP from US Patent database.

2. Study on data searching in LDAP

This study on data searching in LDAP is a part of the above-mentioned study on LDAP. The study is mainly based on the analysis of US patents. The general objectives, methodology and general findings are mentioned in the above-mentioned article. The details of this part of the study on LDAP searching are as follows.

2.1 Objectives of the study

The objective of the study is to know:

- What are the inventions made on data searching in LDAP and on which aspects of it?

- What is the Ideal Final Result (IFR) in LDAP data searching? Is there any trend in the series of inventions?

- Which problems on LDAP data searching are not solved yet? In other words which problems need to be addressed in future?

- Which aspects of LDAP data searching are yet unexplored? What are the possible areas of improvements in future inventions?

- Can we use TRIZ to analyze the patents on LDAP data searching? Which Inventive Principles or other techniques help us getting the solutions?
2.2 Major areas of Invention

- One major area that attracts the inventors is to improve the speed of searching, by implementing various mechanisms.

- Another important area of invention is to improve the cache mechanism for LDAP so that the queries are responded from the cache instead of being executed in the database.

- Another area of invention is to convert LDAP filters to SQL queries or queries in other languages.

- Searching directory data in RDBMS using SQL. Improving LDAP query language.

- Searching and caching semi-structured data, XML data, ACL data and various other types of data in LDAP.

- The other area that attracts the inventors is to create special tables, such as index tables, catalog tables etc. in order to increase the efficiency in data processing, searching and other activities.

- Yet another area that attracts inventors is to find the compatibility of different types of databases used for LDAP storage. This involves conversion from one type of storage to other, data transfer and replication etc. basically by using plug-ins, translators, brokers, data independent formats etc.

2.3 Patents analyzed for the study

- US Patent 6748374, “Method for generating a relational database query statement using one or more templates corresponding to search conditions in an expression tree”,

- US Patent 6199062, “Reverse string indexing in a relational database for wildcard searching”,

- US Patent 6347312, “Lightweight directory access protocol (LDAP) directory server cache mechanism and method”,

- US Patent 6356892, “Efficient implementation of lightweight directory access protocol (LDAP) search queries with structured query language (SQL)”,


Inventions on LDAP data searching- A TRIZ based analysis, by Umakant Mishra
3. Introduction to the Searching Mechanism in LDAP

Searching is one of the most basic operation provided by an LDAP server. LDAP offers a rich set of searching capabilities. The search operation allows a client to request the server to perform a search on its behalf. The search operation locates specific users or services in the directory tree. The LDAP server executes the search (or update) command and returns a response to the client.

According to the LDAP specifications an LDAP search request may contain the following parameters (RFC 2251):
- `baseObject` - an LDAP distinguished name based on which the search is to be performed.
- `scope` - an indicator of searching scope.
- `derefAliases` - how to handle aliases in the searching.
- `sizelimit` - maximum number of entries to be returned in the search result.
- `timelimit` - maximum time allowed for a search.
- `typesOnly` - whether the search results will contain both attribute types and values or just attribute types.
- `filter` - a filter that defines the conditions of the search.
- `attributes` - list of attributes to be returned from each entry which matches the search filter.

Upon receipt of a Search Request, a server will perform the necessary search of the DIT. The server will then return to the client a sequence of responses in separate LDAP messages. The search responses may contain `SearchResultEntry` (the attributes and values satisfying the search conditions), `SearchResultReference` (URL of the referral servers), `ExtendedResponse` or `SearchResultDone` (success or failure of search operation) data types.

steps of an LDAP session

```
Steps of an LDAP session
Initialize LDAP Session
  Open Connection
  Authentication
  Directory Server Operation
    Return Results
    Close Session
```
LDAP servers provide referral services. A referral is a redirection that the directory service returns when the client requests a directory entry that does not exist on the local server. This referral will list servers within the network that contain information that the client is seeking. If the client wishes to progress the search operation, it must follow the referral by restructuring the query and sending it to any of the referral servers.

According to the LDAP specifications, some LDAP servers may hold cache or shadow copies of the entries, which can be used to answer search and comparison queries, but will return referrals or contact other servers if modification operations are requested (RFC 2251).

4. Major Concerns in LDAP data searching

- Searching data in a large directory spread across multiple LDAP servers in a distributed environment may need more time than a user can afford.

- Faster searching increases server load and network traffic. An overloaded server and congested network slows down the performance. This makes a loop of contradiction. It is necessary to reduce server load and reduce network traffic while increasing speed of searching.

- Customizing search options, query optimization, partial searching, wildcard searching, searching upto a defined depth etc. are important issues in LDAP data searching.

- LDAP v3 provides referral service, which means the server returns the references to other servers when the data is not found in the current tree. But the how to customize referral search is left to the vender to decide.

- The LDAP server may store data in a Flat file, RDBMS or any other database. When the data is stored in a relational database, the LDAP search filters need to be converted to SQL formats in order to be executed.

- Although LDAP specifications permit a cache to be implemented it does not specify how to implement it. The actual mechanism of cache remains a concern for the vendors.

5. IFR for LDAP data searching

According to TRIZ, ideality is a function of its benefits and costs. Ideality = Σ Benefits / ( Σ Costs + Σ Harm). So the Ideal Final Result in LDAP searching can be achieved by increasing all useful functions of the LDAP searching and decreasing all harmful functions of the same. An analysis of the LDAP system may find the following as IFR for LDAP administration.
<table>
<thead>
<tr>
<th>Should have (Positive features)</th>
<th>Should not have (Negative features)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• The LDAP server should be able to search the requested entries based on any given condition.</td>
<td>• The LDAP server should never become overloaded.</td>
</tr>
<tr>
<td>• If the result is not found on the local directory, the server should search other directories and respond.</td>
<td>• The user should not enter his credentials again and again for authentication to different LDAP servers.</td>
</tr>
<tr>
<td>• The user should be able to search multiple LDAP servers (though LDAP referral or other means).</td>
<td></td>
</tr>
<tr>
<td>• The search condition may contain full strings or partial strings with wildcard (*) characters.</td>
<td></td>
</tr>
<tr>
<td>• The same commands for LDAP searching should work efficiently in any type of database, flat file, relational database or other.</td>
<td></td>
</tr>
<tr>
<td>• The search operation should be very fast and consume very less (or no) time.</td>
<td></td>
</tr>
<tr>
<td>• The LDAP queries should be automatically optimized and converted to the desired query language to suite the nature of backend database.</td>
<td></td>
</tr>
</tbody>
</table>

6. Inventions on data Searching in LDAP

As searching is the most prominent operation in LDAP, we can find quite a lot of inventions made on this field to improve various aspects of searching. Some of the important inventions on this topic are analyzed below.

6.1 Generating SQL query statements for LDAP Search Filter

Background problem

Although RDBMS is a proven system of data storage and provides many benefits when implemented for large-scale data storage, it is not really suitable for efficient storing of hierarchical LDAP data. The object-oriented data are based on a fundamentally different data model than the relational data. In order to store object orient data in RDBMS, it is necessary to create separate tables for each object class in the system, which causes waste of system resources. Besides it
makes the system rigid, as each change in the object class requires a modification in the database schema.

There is a need for an improved method for storing, searching and updating object-oriented LDAP data in an RDBMS. There is also a need for converting LDAP filters to SQL queries to be executed on RDBMS.

**Solution provided by US Patent 6748374**

US Patent 6748374 presents a method of maintaining object-oriented data in a relational database. This storage mechanism is based on the concept of a previous invention (Patent 6587856) by the same inventors, which propose to use three special tables, viz., “distinguished_name” table, “attribute_store” table and “catalog” tables to facilitate storage and processing of object oriented data in RDBMS. The use of catalog tables provides significant advantage in data searching. As a single catalog table indexes a particular attribute type, it is necessary to search only one single table to query against any particular object attribute type.

Besides, the invention provides a method for automatically generating a SQL query statement to search for particular entries in the DIT stored in RDBMS. The invention can generate only one single SQL statement for any LDAP search filter. This reduces the number of requests and thereby reduces the system and network overload to perform the search.

The invention utilizes a series of templates to convert an arbitrary LDAP search filter into a single SQL statement.

**TRIZ based analysis**

The invention converts LDAP search filters to SQL queries (Principle-36: Conversion), which can be executed on RDBMS.

The invention tries to convert any complicated search filter into only one single SQL statement (Principle-40: Composite), which reduces system overload.

The invention proposes using three intermediary tables including a catalog table, which increases the speed of searching (Principle-24: Intermediary).

**6.2 Reverse indexing LDAP database for faster searching**

**Background problem:**

Lightweight Directory Access Protocol (LDAP) provides lot of query functions such as query searches and updates. Normally the database is indexed (forward indexing) for faster retrieval of data during searches. Although a forward index speeds up most searches it is not very efficient in searches beginning with a
wildcard "*" (e.g., the string "*something"). There is a need exists for an improved method of wildcard searching in a relational database used for LDAP.

**Solution provided by patent 6199062**

US Patent 6199062 discloses a method of wildcard searching in an LDAP directory having a relational database management system (RDBMS) as a backing store. The relational database normally includes a forward index of the character strings in the database. The new method generates both a forward index as well as a reverse index of the key fields in the relational database.

The method examines the position of the wildcard in the search string. Depending upon the position the wildcard in the search string, the invention decides whether to use forward index or reverse index or both the indices to speed up the search.

**TRIZ based analysis**

Wildcard searching should execute very fast on relational database used for LDAP backing store (Ideal Final Result).

The invention uses a reverse indexing mechanism that improves performance in wildcard searching (Principle- Other way round).

Use forward index if there is no wildcard (or the wildcard is trailing after a certain number of characters). Use reverse index, when there is a wildcard at the beginning of the search string (or trailing before a certain number of characters) to speed up the search (Principle-15: Dynamize).

**6.3 LDAP directory server cache mechanism**

**Background problem:**

The LDAP server executes the queries and sends the responses back to the LDAP clients. In practice there are lot of repetitive queries to the LDAP that takes up a lot of time to execute the same queries again and again. There is a need for efficient handling of repetitive searches issued from a hierarchical directory service to a backend database system.
Solution provided by US Patent 6347312

US Patent 6347312 discloses a method of faster searching of LDAP data in RDBMS by using a cache. According to the invention when LDAP service sends searches to the backend database, the results of the searches are cached in the directory service. Instead of depending on the cache of the database service, the mechanism populates a local storage of cache associated with LDAP directory service.

The invention prefers to use two caches. The first cache (named Type-I cache) receives a set of identifiers indexed by a filter key of the search query. The second cache (named Type-II cache) stores the search results or the entries corresponding to the set of identifiers.

When the same queries come again and again, the LDAP server serves the results from the cache without even sending requests to the backend database. This saves the query execution time and makes the response faster. When an entry is updated in the directory data the corresponding cache is invalidated to maintain integrity of the cached information.

TRIZ based analysis:

The LDAP server should be efficient to respond to clients’ queries very fast (or within no time) (Ideal Final Result).

LDAP server should not re-execute the same queries again and again on client’s request that are already executed previously (Desired Result).

Let the server store the result of previously executed queries (in a cache) and use them for successive requests (Principle-10: Prior Action, Principle-26: Copying, Principle-27: Cheap and disposable).

The invention proposes to use two sets of caches, viz., type-I cache and type-II cache (Principle-1: Segmentation).

When the same queries come again and again, the LDAP server serves the results from the cache (Principle-3: Local Quality) instead of sending requests to the backend database.
When an entry is updated in the directory data, the corresponding cache is invalidated. The cache is updated again in the next query on that entry (Principle-34: Discard and recover).

### 6.4 Implementing SQL in LDAP queries

#### Background problem

The query functions are used very frequently in an LDAP environment. The functions include searching, filtering, comparing and updating etc. These functions are typically very slow in LDAP environment. The similar queries through SQL run very fast on RDBMS. It is necessary to implement LDAP search queries in SQL in order to use RDBMS for LDAP storage.

Although LDAP and SQL use the same AND, OR and NOT logical operators, it is possible to translate LDAP filters to SQL queries. But there is no known method that works well in practice. There is a need for efficient mapping of an LDAP filter into an SQL query.

#### Solution provided by US Patent 6356892

US Patent 6356892 discloses a method of converting LDAP filter expressions to an SQL query to retrieve the matches from the backend database. The method parses an LDAP filter-based query to find the elements and logical operators and maps the LDAP filter efficiently into an SQL sub-query. Then the SQL sub-queries are combined into a single SQL query according to a set of combination rules.

![Diagram](diagram.png)

#### TRIZ based analysis

The invention generates a SQL sub-query for each LDAP operator based on given translation rules (Principle-28: Mechanics Substitution).

The SQL sub-queries are then combined into a single SQL query according to a set of combination rules (Principle-5: Merging, Principle-40: Composite).
6.5 Multi-level directory searches in LDAP

Background problem:
An LDAP server is queried by LDAP clients, which creates an LDAP search to the LDAP database and provides the results to the clients. The search request may include a scope that specifies the depth of the search in the directory hierarchy. As LDAP does not specify a particular storage mechanism, it can use flat files, a binary tree or a relational database for back end storage. But the searching scope is limited when a relational database is used as the backing store for LDAP. There is a need for a mechanism that can support searching up to any level in a directory tree implemented in relational database.

Solution provided by patent 6625615:
US Patent 6625615 discloses a method of performing a multi-level directory search in the relational database. An ancestor table is generated which maps each node in the directory that is an ancestor of one or more descendant nodes to each of its ancestor nodes. Each node is associated with a unique identifier, which may be used to relate ancestor and descendant nodes. The table includes the identifier of each ancestor in association with all the corresponding descendants. Additionally, the table maps the distance between the ancestor and each descendant. When a search request comes with a search scope parameter the distance values are used to delimit the search up to that distance.

<table>
<thead>
<tr>
<th>AEID</th>
<th>EID</th>
<th>DISTANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

TRIZ based analysis
The invention uses an ancestor table that maps each node in the directory in relation to ancestor and descendant nodes, which is used to compare the depth and delimit the search operation (Principle-10: Prior Action, Principle-24: Intermediary).

6.6 Scalable event notification in LDAP systems

Background Problem:
A persistent search is an ongoing search that provides a mechanism by which an LDAP client can receive notification of modifications in an LDAP database. Each persistent search maintains an open TCP/IP connection to the LDAP server. A persistent search does not end after an initial set of entries matching the search criteria of the persistent search of the client have been returned. Rather, the
LDAP server continues the persistent search via an active channel through which entries that are modified, as well as additional information about the modifications that occur, can be communicated.

When the number of persistent-search clients increases, the performance of the LDAP server suffers to a great extent. There is a need to provide a more efficient, less resource-intensive, and faster system and method to perform directory searches of LDAP servers.

**Solution provided by patent 6769011:**

US Patent 6769011 describes a method of more efficient, less resource-intensive and faster directory searches in LDAP servers. According to the invention, a portion of the directory server is searched using a proxy, and the proxy also gets registrations from the clients. The proxy can combine registrations from multiple clients into a single search of the LDAP directory. Registrations by the clients can be restricted to the portion of the LDAP directories searched by the proxy.

The LDAP servers notify the proxy concerning modifications to entries in the LDAP directories. In response to a determination that modified entries in the LDAP directories matches registration criteria received from one or more clients, the proxy notifies the appropriate clients of the modified entries.

![Image](image.png)

**FIG. 3**

**TRIZ based analysis**

According to the invention, the directory server is searched through a proxy. The proxy gets registrations from the clients. The modifications to the entries are also communicated to the clients through the proxy (Principle-24: Intermediary).

The proxy combines registrations from multiple clients into a single search to save execution time (Principle-40: Composite).
6.7 Method of caching LDAP queries

Background problem:
To improve the response and availability of directory services, it is desirable to be able to cache results of directory queries and to use the cached results for answering future queries. However, unlike other Web content, individual resources (entries) within LDAP directories are not accessed directly but instead use LDAP queries. Therefore, techniques used for Web page caching cannot be used for LDAP resources. For an LDAP cache to answer a query, it needs to check whether the query is semantically contained in earlier queries. There is a need exists for providing caching solution for LDAP queries, particularly which can efficiently check the containment of new queries in the previous queries.

Solution provided by US Patent 7035846
US Patent 7035846 provides a method for processing directory queries in LDAP through a proxy server. The proxy server is configured to receive the directory queries from the clients. The proxy server maintains a history of queries and results in the cache. It compares the query characteristics stored in the cache and determines whether the query can be answered from the cached queries. If the query (or subsets of the query) is found to be there in the proxy cache, the proxy server retrieves the data from the cache and responds to the client, otherwise it sends a request to the LDAP server. The new result is sent to the client and the new query and the resulting data is also added to the cache for future reference.

TRIZ based analysis
According to the invention the proxy server maintains a history of queries and results in the cache (Principle-10: Prior Action).

Depending on the query characteristics, if the proxy finds the answers in the proxy cache, it responds the clients with the data from the cache (Principle-3: Local Quality).

If the proxy does not find the answers in the proxy cache it sends a request to the LDAP server and the client is responded from the new result (Principle-11: Cushioning).
6.8 Authentication referral search for LDAP

Background problem
LDAP protocol provides a referral mechanism, by which the LDAP server, upon receiving a search query from LDAP client, may return the answer to that query or may return referrals to other servers that may contain the requested information. The client computer automatically reformats the original LDAP request and reissues the request to contact the referral servers. The contacted servers, in turn, might return the requested information or might return the name of other servers, which might contain the requested information.

One of the major problems associated with this referral mechanism is that the user needs to bind to other servers, with different Distinguished Names (DN's) existing on these servers. Without this binding, the referred search becomes an unauthenticated request. Therefore, it would be desirable to have a method, which allows a user to manage information stored on all servers without having his or her account physically reside on every server.

Solution provided by patent 7016897
US Patent 7016897 provides a method of authenticating LDAP referral searches. According to the invention, the LDAP server receives a bind request from the LDAP that referred the search request and then searches the local directory for an entry corresponding to the Distinguished Name (DN) of that bind request. If the DN is there on the server, then the server performs the authentication of the referred search request. If the DN is not located on the server, the server then checks the defined reference server. If the DN is located in the reference server, the reference server is contacted for authentication. If the bind DN is not found in the reference server the request is not authenticated and is denied.
TRIZ based analysis

According to the invention, when the LDAP server does not have the DN, it contacts the other reference servers to authenticate (Principle-27: Cheap and short living objects).

7. Other related inventions

7.1 Data mapping in RDBMS for searching LDAP data

US Patent 6085188 discloses a method of organizing hierarchical LDAP data in a relational database used as a backing store for LDAP. The invention proposes to add two more relational tables in the LDAP database (Principle-24: Intermediary). The LDAP server does a recursive search beforehand and stores the unique identifiers of all the parents and children in these two tables (Principle-10: Prior Action). The query searches these relationship tables first instead of doing a recursive search in the whole database. Using these two tables of parent child relationship enhance the performance of searching.

7.2 Invention-13: Automated distinguished name lookup

It is often difficult for the user to remember the unique distinguished names to feed the LDAP server for a searching request. US Patent 6408306 discloses an automated distinguished name lookup system. According to the invention, the user can enter a part of the distinguished name using a wild card format and the system displays all the matching names to the user (Principle-16: Partial or excessive action). The user then selects the distinguished names as desired for LDAP directory operation.

7.3 LDAP-based distributed cache technology for XML

US Patent 6901410 discloses a method dealing semi-structured data in the LDAP environment using XML. The invention has three steps, (i) transforming the semi-structured data into LDAP data by converting the semi-structured data as the attributes of individual nodes; (ii) converting a query, written in a semi-structured query language for operation on said semi-structured data, into an LDAP query; and (iii) accessing said LDAP data with said LDAP query (Principle-28: Mechanics Substitution, Principle-36: Conversion).

According to the invention, a query, written in a semi-structured language, can be naturally split into a sequence of sub-queries; it becomes easy to cache the results of sub-queries independently. The process of evaluating sub-queries are simpler then complete queries and easy to manage in the cache (Principle-1: Segmentation).
7.4 Searching profile attributes based on other target profile attributes and associated profiles

US Patent 6470332 discloses a method of searching and retrieving a profile (or directory) attribute based on the other attributes of the same profile or other attributes of other associated profiles. The invention uses string search syntax according to RFC 2254, but it is enhanced to allow multiple related search filters to be specified at one time (Principle-40: Composite). The top-most filter is used to retrieve results and the succeeding filters are used to determine if a specific profile should even be considered.

8. Summary and Findings of the study

8.1 Distribution of patents

- A total of twelve patents are presented in the study. Out of them eight patents were found to be more relevant to LDAP data searching, which are analyzed in detail. The rest four patents are analyzed in nutshell.

- Out of the 12 patents illustrated, 8 patents are assigned by International Business Machines Corporation, one assigned by Oracle corporation, one by Sun Microsystems and one by Telefonaktiebolaget LM Ericsson.

8.2 Hot areas in LDAP data searching

The study of patents on LDAP searching mechanism reveals the following areas, which are quite hot, and needing improvements.

- Converting LDAP filters to SQL queries to execute on relational database.

- Implementing proxy and responding queries from proxy cache.

- Implementing LDAP cache and responding from LDAP cache.

- Improving search filters, customizing search options, query optimization, improving search speed in larger environments.

- Improving data searching and updating in distributed environment.

- Improving and customizing LDAP referral service.

- Adding intermediate reference files for faster searching.

- Customizing depth of searching in large distributed environment.
8.3 Trends of evolution in LDAP storage

The analysis finds the following trends to be more visible among the inventions on LDAP data storage.

- Increasing speed in LDAP data searching- Inventions want increase the speed of searching by implementing various mechanism.

- Increasing easiness to “configure and implement” and increasing easiness to “maintain and administer” are two obvious trends.

- Increasing Dynamization- Inventions try to provide flexibility and customization in data searching.

- Reducing Human Involvement- Inventions try to reduce human involvement in language conversion and query execution.

- Reducing system complexity- The LDAP query language should remain simpler for easy learning by the users.

- Mono-bi-poly- the scope of searching increases from one LDAP server to multiple LDAP servers and from one directory tree to multiple directory tree.

- Boundary breakdown- searching scope tends to go beyond LDAP servers to include other types of directory servers in the network.

- Reducing conversion- the LDAP queries should run efficiently on the back end database with minimum conversion.

8.4 Predicting future inventions on LDAP searching

Based on the analysis we can predict more inventions on the following aspects of LDAP data searching in future.


- Efficient conversion of LDAP search filters to SQL queries (Principle-36: Conversion).

- Improving SQL implementation in RDBMS used as LDAP backing store (Principle-38: Enrich).

- Improving speed of searching by accessing the nearest replication based on geographical location.
- Searching data from external Directory Information Trees (DIT) using LDAP queries (Principle-17: Another dimension).
- Dividing LDAP cache for faster access (Principle-1: Segmentation).
- Method of searching object oriented ACL data, semi structured data and other types of data in LDAP queries.
- Hierarchical caching model for LDAP database (Principle- Partial or excessive action).

Reference to patents:


Other references:

13. Umakant Mishra, Inventions on LDAP- a Study Based on US Patents
   http://works.bepress.com/umakant_mishra/52


15. Umakant Mishra, “Inventions on LDAP data storage- a TRIZ based Analysis”,
   http://papers.ssrn.com/abstract=925921


17. Umakant Mishra, Inventions on LDAP Administration- A TRIZ Based Analysis (August 2006).
   Available at SSRN: http://ssrn.com/abstract=925932


