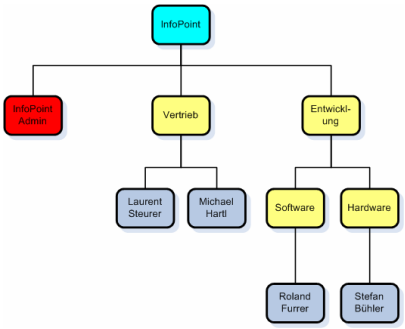
LDAP

Lightweight = L
Directory = D
Access = A
Protocol = P

Das Client-/Server Protokoll für Verzeichnisdienste



Michael Hartl, Laurent Steurer

8. Februar 2006

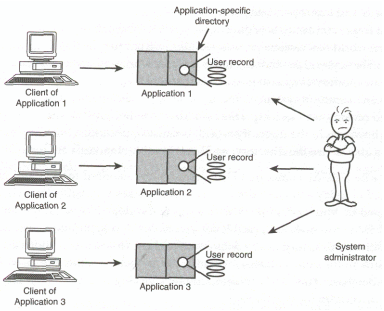
Inhalt

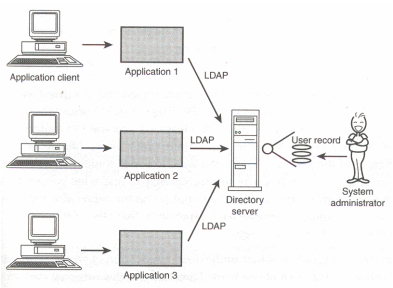
- Motivation Verzeichnisdienste
- Verzeichnisdienste
- Geschichte
- Was ist LDAP?
- LDAP vs. Datenbanken
- Produkte
- LDAP-Modelle
- LDAP-Schema
- LDIF (LDAP Directory Interchange Format)
- Active Directory
- Single Sign-On
- Beispiel I: Outlook
- Beispiel II: Geburtstags-Reminder
- Fragen und Quellen

Motivation Verzeichnisdienste

- Wie lautet die Email-Adresse des Projektleiters?
- Welches ist der nächstgelegene Drucker?
- Einfaches Teilen des Zugriffs auf Adressbuchs mit Handelspartnern in einer Extranet-Umgebung
- Zentrales Telefonbuch
- Zentrale Email-Verwaltung
- Keine veralteten Daten, stets aktuell
- Zentrale Anmeldung ('Single Sign-On')
- Zentrale Verwaltung der Daten aus unterschiedlichen Bereichen

Motivation Verzeichnisdienste II



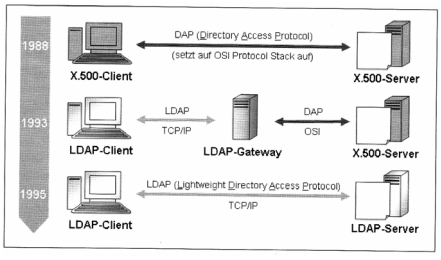


Verzeichnisdienste

- Ein Verzeichnisdienst kann man sich als eine spezialisierte Datenbank vorstellen.
- Verwaltet Informationen über Objekte (Computer, Programme, Dienste) und deren Eigenschaften (Attribute) in einem verteilten Computersystem oder dem Internet.
- Anwender können Objekte mithilfe des Verzeichnisdienstes auffinden und verwenden
- Administratoren können die Objekte verwalten
- Applikationsübergreifende Daten sollen zentral abgespeichert werden (Vermeidung von Daten-Redundanzen)

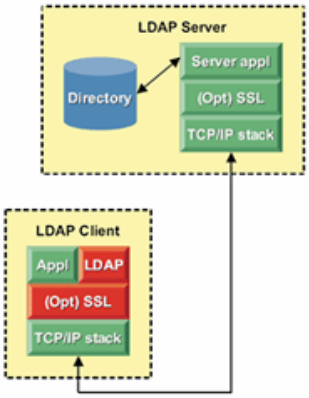
Geschichte

- Am Anfang war
  - X.500 mit DAP (X.500s directory client access protocol)
  - DAP - umfangreich, komplex, schwierig zu Implementieren, X.500/OSI Protokollstack
- LDAPv1: 1993
  - beide mit TCP/IP und Gatewayrechner zwischen Tcp/Ip und X.500
- LDAPv2: 1995
  - erste breit eingesetzte Version
  - setzt durchgängige auf TCP/IP auf
- LDAPv3: 1997 - 2000
  - Sicherheit - Stichwörter: SASL, TLS, SSL
- LDAPv4: scheint in weiter Ferne
  - zuerst soll LDAPv3 von RFC zu einem Standard werden



Was ist LDAP?

- ist ein effizientes auf TCP/IP basiertes Kommunikationsprotokoll für Verzeichnisdienste
- eine normierte Sprache wie LDAP Client + Server miteinander kommunizieren
- LDAP vs. X.500 ist ein schlankes (lightweight), effizientes, geradliniges Protokoll, welches einfach zu implementieren ist
- Client/Server Protokoll wie:
  http um auf Webseiten zuzugreifen
  imap um auf Mails zuzugreifen
- LDAP ist **kein** textbasiertes Protokoll, deshalb kann man nicht wie bei http, pop, imap, smtp einfach mittels Telnet zugreifen
- Standardisierte Daten-Schemen gewährleisten Kompatibilität zwischen LDAP-Anbietern



LDAD vs. Datenbanken

- **Verzeichnisdienste sind auf Lesevorgänge optimiert**
  - Optimiert auf schnelles Finden und Lesen
- **Verzeichnisdienste können einfacher erweitert werden**
  - kennen viele vordefinierte Datentypen (people, organisation, etc).
  - neue Datentypen mit neuer Semantik können einfach eingeführt werden
- **Verteilte Datenhaltung (Distribution scale)**
  - jede Aussenstelle verwaltet die Daten auf eigenem LDAP-Server
  - gegen aussen werden alle zu einem grossen Baum zusammengeschaltet
- **Daten Replizierung (Replication scale)**
  - bei DBs ist das sehr aufwändig und nur für begrenzte Anzahl Replicas möglich (wegen der strengen Konsistenz-Bedingungen die einzuhalten sind)
  - Bei Verzeichnisdiensten spielt es nicht so eine Rolle, wenn die mal nicht ganz gesyncht sind.
- **Performance**
  - typischer Verzeichnisdienst: 10‘000 Queries / Sekunde
  - typisches DB: 100 Transaktionen / Sekunde (Transaktionen sind komplex)
- **Standards und Interoperabilität**
  - LDAP liefert die Standardmodelle und Protokolle für Verzeichnisdienste
  - **Standardisiert:** ein LDAP-Client von Microsoft arbeitet mit einem LDAP-Server von Netscape
- **Joins**
  - gibt es nicht
  - Erfordert mehrfaches traversieren des Baums (Referenzen über DN‘s)

Produkte

- OpenLDAP
- Microsoft Active Directory
- ADAM (Active Directory Application Mode) (abgespecktes Active Directory für Desktop-Betriebssysteme)
- Novell eDirectory
- Netscape Directory Server
- Oracle Internet Directory
- Lotus Domino
- IBM Directory Server

LDAP-Modelle

Besteht aus vier Modellen:

- **Informationsmodell** - beschreibt was in ein Verzeichnis abgelegt werden kann
- **Namensmodell** - beschreibt wie Daten angeordnet und angesprochen werden
- **Funktionsmodell** - beschreibt was man alles mit Daten machen kann (search, delete, add, …)
- **Sicherheitsmodell** - beschreibt wie Verzeichnisse geschützt werden können

Informationsmodell

- Entry (vergleichbar mit Klasse unter Java)
  - Sammlung von Informationen über ein Objekt, häufig identisch mit der realen Welt
  - ein Entry besteht aus:
    - eindeutige Identifizierung (DN = distinguished name)
    - Bündel von Attributen
- Attribute
  - jedes Attribut beschreibt eine besondere Eigenschaft des Objektes
  - ein Attribut besteht aus:
    - ein oder mehrere Values
    - Syntax Definition - wie müssen Values "ausschauen" (Datentyp)
    - Regeln für Vergleich von Values (z.B. ignoriere Gross-/Kleinschreibung)
    - Regeln für Sortierung (z.B. lexikalisch, numerisch)
    - kann optional oder zwingend sein
  - Unterscheidung in zwei Typen:
    - user attributes – „normale" Daten durch Nutzer gepflegt
    - operational attributes – werden automatisch gepflegt z.B. modifyTimeStamp

Namensmodell

- wie Dateisystem, jedoch:
  - es gibt keinen richtigen Root Eintrag (vgl. / im Dateisystem)
  - jeder Knoten kann auch Container für Daten sein (Verzeichnisse können keine Daten aufnehmen)
  - Leserichtung bei Namensgebung (Dateisystem Wurzel → Blatt)
- kein allgemein gültiges Muster für einen Baum, → individuelle Ausarbeitung erforderlich
- bestehende Baum-Struktur schwierig zu ändern, da die Pfade in den Anwendungen häufig "hartcodiert"

- **DN = Distinguished Name**
  Absoluter, eindeutiger „Pfad" des Objekts im Baum
  (Beispiel: cn=admin,o=System,dc=r-klaproth,c=de)
- **Base DN = Einstiegsknoten** (Wurzel)
- **RDN = Relative Distinguished Name**
  Relativer „Pfad" ausgehend vom übergeordneten Knoten
  (Beispiel: cn=admin)

Funktionsmodell

- Abfrage Operationen
  - Search (zentrale Funktion bei Directories)
  - read
  - compare
- Update Operationen
  - add
  - delete
  - modifiy
  - modifiy DN (rename)
- Authentifizierungs & Kontrolloperationen
  - bind - Authentifizieren
  - unbind - Abmelden
  - abandom - Anfrage abbrechen

Sicherheitsmodell

- Informationen vor unbefugtem Zugriff schützen
- ACLs (Access Control Lists) regeln den Zugriff bis auf Attributebene
- LDAPv2 unsicher (Passwort als Klartext)
- LDAPv3 bietet SSL-, TLS-Verschlüsselung
  - unterstützt Framework „Simple Autentication and Security Layer (SASL)" um verschiedenste Authentifizierungs-Methoden einzubinden

Schema: LDAPv3 (RFC 2256)

- Standardisierte LDAP-Schemas gewährleisten Kompatibilität zwischen Verzeichnisdienst-Anbietern
- abstrakte Objektklasse „top" bildet die Wurzel der Ableitungshierarchie
- Child-Klassen enthalten automatisch alle Attribute ihrer übergeordneten Parent-Klasse
- Vordefinierte Objekte (objectClass): 22 Stück
  - organization, person, organizationalRole, …
- Vorderfinierte Attribute: 55 Stück
  - cn, telefonNumber, mail, …

Schema: LDAPv3 (RFC 2256)

- **Objectclass-Syntax:**

  objectclass (ObjectID NAME name [DESC desc] [SUP sup] [ABSTRACT | STRUCTURAL | AUXILIARY]
  [MUST must] [MAY may])

```
objectclass ( 1.3.6.1.4.1.4203.666.100.1
     NAME 'officePerson'
     DESC 'Office employee or computer user'
     SUP inetOrgPerson
     STRUCTURAL
     MUST ( o $ phone )
     MAY  ( c $ comment )
)
```

- **Attribute-Syntax:**

  attributetyp (AttributID NAME name [DESC desc] [SUP sup] [EUQALITY equality] [ORDERING ordering]
  [SUBSTR substr] [SYNTAX syntax] [SINGLE-VALUE])

```
attributetype ( 1.3.6.1.4.1.4203.666.100.129
     NAME ( 'birthday' )
     DESC 'Geburtstag als Datum,
     SUP name
)
```

Legende:
[] = optionaler Wert
 = Schlüsselwort

Schema: Anwendungsbezogen

- Regelwerk:
  Welche Datenarten dürfen gespeichert werden?
  → definiert die erlaubten Objektklassen
  → definiert die erlaubten Attribute
- Sicherstellung von Integrität und Qualität der Daten
  → Daten können nur eingefügt werden,
  wenn sie die Schema-Definition erfüllen
- sind auf dem Server hinterlegt
  (in der Regel als Text-Dateien)

LDIF (LDAP Directory Interchange Format)

- Format zum Austausch von Daten verschiedener LDAP-Anbieter
- Text-Format
- Objekte einer Baumstruktur müssen beginnend von der Wurzel beschrieben werden

Beispiel:

```
dn: cn=Laurent Steurer, ou=Vertrieb, o=InfoPoint
objectClass: top
objectClass: person
objectClass: officePerson
cn: Laurent Steurer
mail: l.steurer@gm.ch
telefonNumber: 071 245 95 71
```

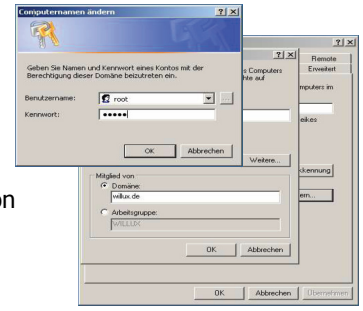
Active Directory

- vereinigt (unnötigerweise) drei Services
  - LDAP-, DNS-, Kerberos-Server
- mit Windows 2000 Server erstmals eingeführt
- löst Windows-Benutzerdatenbank aus NT-Zeiten ab
- NOS-based / General-Purpose Service
- nach Installation vorgegebene Struktur mit vielen Informationen bereits vorhanden
- sehr einfacher Zugriff mittels .NET
  - siehe Namespace System.DirectoryServices
- nicht alle Attribute zugänglich (z.B. Passwort des Users)
- Exchange Server legt z.B. seine Daten im Active Directory ab
- Teilbäume heissen Domänen (Struktur nach Abteilung, Standort, …) mehrere Domänen bilden den Forest
- Microsoft verschleiert/verwässert offene Standards
  - proprietäre Protokollerweiterungen von LDAP
  - Propagiert .NET anstelle LDAP-API

Single Sign-On

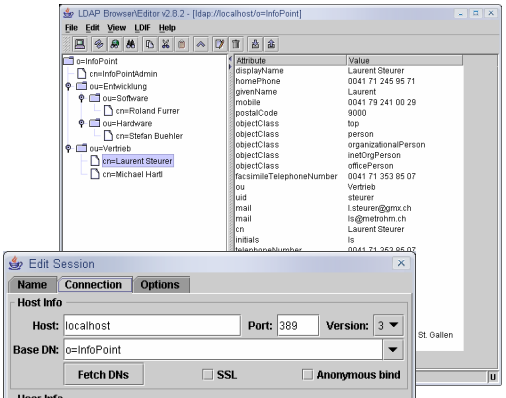
- **Zentrale Anmeldung:**
  Man meldet sich nur einmal am Netzwerk an und ist damit gegenüber allen Diensten authentifiziert
- Beispiele für Dienste:
  OpenLDAP, Samba (Fileserver), Apache WebServer, …

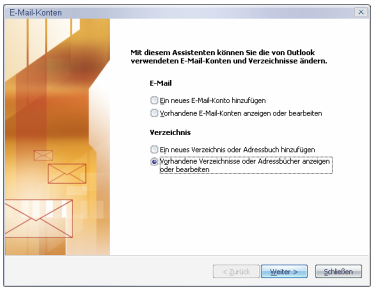
- Voraussetzung:
  - Authentifizierungsmechanismus (defacto Kerberos 5)
  - LDAP: verwaltet Benutzer-Accounts, liefert Daten (Telefonliste, …)
  - Eingebundene Applikationen müssen Kerberos oder SASL (Simple Authentication and Security Layer) unterstützen



Beispiel I: LDAP u. Outlook

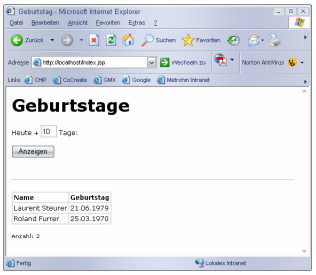
- Verbindungsprofil benötigt:
  - Hostname des Servers
  - Port (Standard: 389)
  - Base DN
  - User-Login





Beispiel II: LDAP u. Java

Geburtstags-Reminder



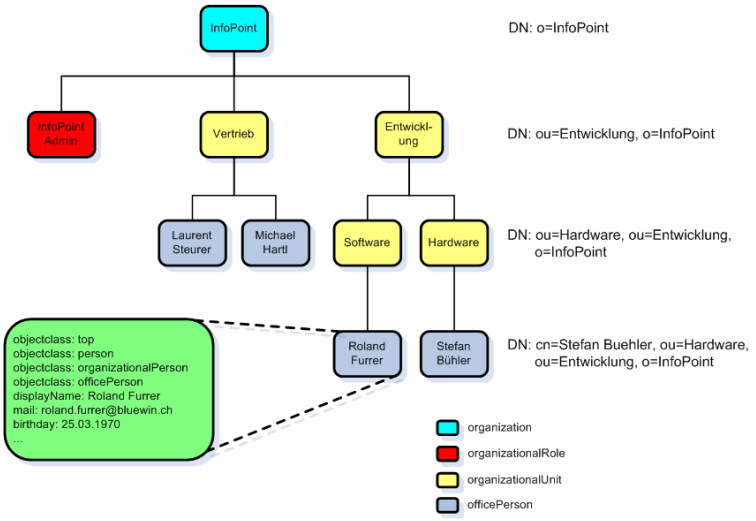

```
private final String user_dn    = "cn=InfoPointAdmin,o=InfoPoint";
private final String password   = "secret";
private final String host       = "ldap://127.0.0.1";
private final int port          = 389;
private final String base_dn    = "o=InfoPoint";

private InitialDirContext rootContext = null;

public Ldap() throws NamingException
{
    String  provider_url = host + ":" + port + "/" + base_dn;

    Hashtable env = new Hashtable();
    env.put(Context.SECURITY_AUTHENTICATION, "simple");
    env.put(Context.SECURITY_PRINCIPAL, user_dn); // User
    env.put(Context.SECURITY_CREDENTIALS, password); // Password
    env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
    env.put(Context.PROVIDER_URL, provider_url);

    // Verbindungsaufbau
    rootContext = new InitialDirContext(env);
}
```

```
public ContextItem[] search(String[] attrIds, String dn, String objectClass, boolean oneLevelScope)
{
    SearchControls ctls = new SearchControls();

    // Such-Tiefe
    if (oneLevelScope) ctls.setSearchScope(SearchControls.ONELEVEL_SCOPE);
    else ctls.setSearchScope(SearchControls.SUBTREE_SCOPE);

    ctls.setReturningObjFlag(true);
    ctls.setReturningAttributes(attrIds);

    // objectClass-Filter
    String filter = "(objectClass=*)";
    if (objectClass != null) filter = "(objectClass=" + objectClass + ")";

    NamingEnumeration enumSearchResult = rootContext.search(dn, filter, ctls);
    // rootContext = LDAP-Connection

    ArrayList data = new ArrayList();
    while (enumSearchResult.hasMore()) {
          SearchResult result = (SearchResult)enumSearchResult.next();
          Attributes attrs = result.getAttributes();
          ...
    }
}


public ArrayList searchPersons(int days)
{
    // Such-Vorgang anhand LDAP
    ContextItem[] items = ldap.search(new String[] {"birthday", "displayName"}, "", "officePerson",
    false);

    if (items != null)
    {
       ...
      // Vergleich von Datum anhand Timestamps
    }
}
```

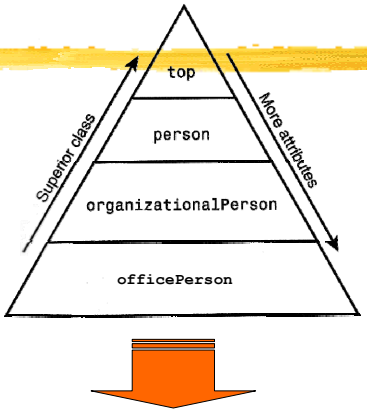
Links und Quellen

- LDAP und Java:
  - LDAP für Java-Entwickler, Stefan Zörner u. Jörg Wegener, Software & Support Verlag GmbH, ISBN 3-935042-58-2
- LDAP-Server:
  - http://www.openldap.org for Linux
  - http://lucas.bergmans.us/hacks/openldap/ for Windows
- LDAP-Browser:
  - http://www-unix.mcs.anl.gov/~gawor/ldap/
  - http://www.ldapbrowser.com/ (Java)
- Weitere Informationen zu LDAP:
  - http://www.mitlinx.de/ldap/main.htm
  - http://de.wikipedia.org/wiki/LDAP
  - http://www.verzeichnisdienst.de/ (Umfangreiche Linksammlung zu LDAP-Verzeichnisdiensten)
  - http://ldap.akbkhome.com/index.php (LDAP-Schema, Syntax, ...)

Namensmodell



Schema: LDAPv3 (RFC 2256)



Attribut	Beschreibung
mail	E-Mail
telephoneNumber	Telefonnummer
cn	common name Name der Person/Objekt

LDIF-Befehle (Kommandozeile)

Allgemeine Parameter:

- **-x** Simple Authentication
- **-D** Bind DN
- **-W** Prompt für Bind-Passwort
- **-h** Host
- **-b** Base-DN (nicht alle Programme)

- **ldapmodify** - Ändert / löscht / ergänzt Attribute von Objekten

```
ldapmodify -x -D "cn=Admin,dc=example,dc=de" -W -f modify.ldif
```

modify.ldif:

```
dn : cn=Karsten Petersen , dc=example , dc=de
changetype : modify
replace : sn
sn : Petersen
```

LDIF-Befehle (Kommandozeile)

- **ldapadd** - fügt Objekte dem Directory hinzu

```
ldapadd -x -D "cn=Admin,dc=example,dc=de" -W -f add.ldif
```

add.ldif:

```
dn : cn=Karsten Petersen , dc=example , dc=de
objectClass : top
objectClass : person
sn : Petersen
cn : Karsten Petersen
```

- **ldapdelete** - löscht Objekt mit angegebenem DN

```
ldapdelete -x -D "cn=Admin,dc=example,dc=de" -W "cn=Karsten
Petersen,dc=example,dc=de"
```

LDIF-Befehle (Kommandozeile)

- **ldapsearch** - Sucht nach Objekten und gibt diese aus
- sehr aufwändige Filterung und Auswahl der Ausgabe möglich

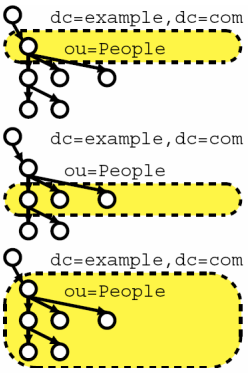
```
ldapsearch -x "(cn=Karsten Petersen)"
```

```
[...]
# Karsten Petersen , example , de
dn : cn=Karsten Petersen , dc=example , dc=de
objectClass : top
objectClass : person
sn : Petersen
cn : Karsten Peterson
[...]
```

Suchen

Parameter für die LDAP Suchanfrage:

- Basis Objekt (z.B. DN des ersten Entries)
- Suchtiefe (base, onelevel, sub)
- Maximale Grösse der Ergebnismenge (0 bedeutet alles bis Server-Maxmimum)
- Zeitlimit in Sekunden
- Suchfilter Objekttypen (objectClass)
- Liste gewünschter Attribute



Verzeichnisdienst Typen

- ***NOS-based directories***
  speziell designed um den Anforderungen von Netzwerk-Betriebssystemen zu entsprechen
  Beispiele: Novell's NDS, Microsofts Active Directory, Banyan's StreetTalk
- ***Application-specific directories***
  Diese Verzeichnissedienst kommen gebündelt oder integriert mit einer Anwendung.
  Beispiele: Lotus Notes Namen und Adress-Buch, Microsoft Exchange, Novell GroupWise
- ***Purpose-specific directories***
  Diese Verzeichnissedienste sind nicht an eine Anwendung gebunden, jedoch nur für einen speziellen Zweck bestimmt und sind nicht erweiterbar.
  Beispiele: Domain Name System (DNS)
- ***General-Purpose, standards-based directories***
  Diese Verzeichnissedienste wurden entwickelt, um die Anforderungen einer breiten Zahl von Anwendungen abzudecken.
  Beispiele: LDAP Verzeichnisdienste und auf X.500 basierende Verzeichnisdienste