

# LDAP

Orientation Réseaux et Services, Module EIP, Unité PDA

Olivier Liechti

Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud (HEIG-Vd)

Yverdon-les-Bains

# Planning

- > **Semaine 1 (21/09)**
  -
- > **Semaine 2 (28/09)**
  -
- > **Semaine 3 (5/10)**
  -
- > **Semaine 4 (12/10)**
  -
- > **Semaine 5 (19/10)**
  -
- > **Semaine 6 (2/11)**
  - LDAP: protocole + infrastructure
- > **Semaine 7 (9/11)**
  - Laboratoire LDAP
- > **Semaine 8 (16/11)**
  - **Travail écrit LDAP**

**délai** : lundi 19 novembre, 8h00

- > **Semaine 7**
  - Laboratoire LDAP
- > **Semaine 8**
  - Travail écrit LDAP

# Agenda

## > Introduction

- Protocole LDAP, objectifs et utilisations typiques, historique
- Outils: serveurs, browsers, APIs et librairies

## > LDAP: le modèle

- Organisation hiérarchique des données, nommage
- Notions: DIT, entrée, attribut, classe, schéma

## > LDAP: le protocole

- Principes de base, opérations, le format LDIF

## > LDAP: l'infrastructure

- Distribution et réplication
- Commandes, filtres de recherche, etc.

## > LDAP avec Java: Java Naming & Directory Interface (JNDI)

- Authentification, recherche, manipulation d'entrées

# Références

- > **LDAP for Rocket Scientists (ZYTRAX, Inc.)**
  - <http://www.zytrax.com/books/ldap/>
- > **Redbook IBM**
  - <http://www.redbooks.ibm.com/abstracts/sg244986.html>
- > **Tutoriels et de présentations**
  - <http://quark.humbug.org.au/publications/ldap/>
  - <http://www.it-sudparis.eu/s2ia/user/procacci/ldap/>
  - <http://www.hawaii.edu/its/brownbags-trainings/ldap/>
- > **Liste des RFCs**
  - <http://www.mozilla.org/directory/standards.html>
- > **Open DS**
  - <https://opens.dev.java.net/>
- > **Clients LDAP**
  - <http://directory.apache.org/studio/>
  - <http://www-unix.mcs.anl.gov/~gawor/ldap/>

# *Introduction*

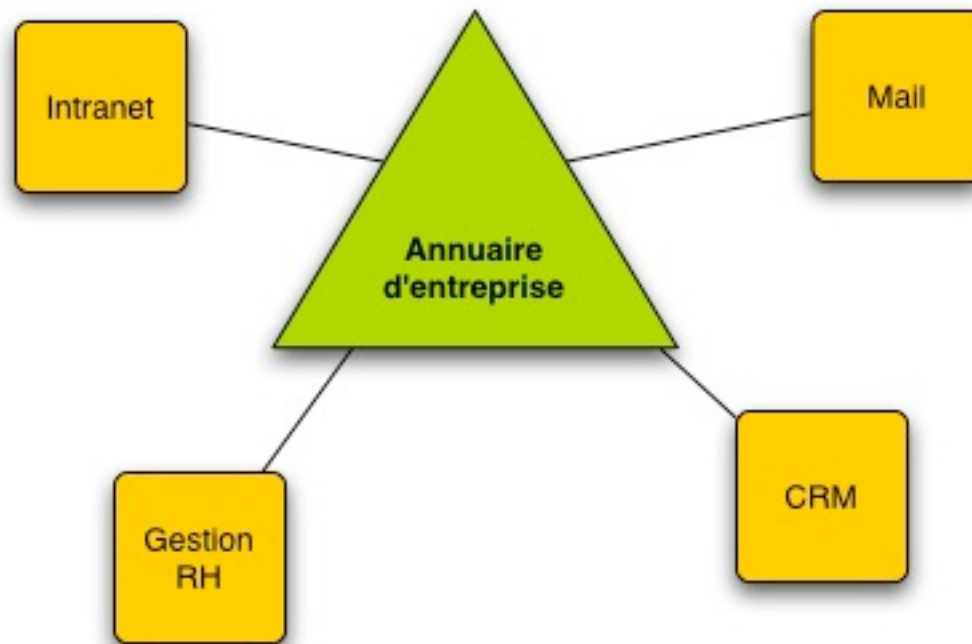
# LDAP: un service d'annuaire

- > **Fin des années 70**
  - Standardisation d'un service d'annuaire par l'UIT (X.500).
  - Développé notamment pour répondre aux besoins du service de messagerie.
  - Directory Access Protocol (DAP).
- > **Début des années 90**
  - Version simplifiée du protocole utilisant TCP/IP.
  - University of Michigan, IETF
- > **Fonctions clés**
  - Recherche d'information rapide
  - Authentification



[http://flickr.com/photos/gehmflor/375334958/sizes/m/#cc\\_license](http://flickr.com/photos/gehmflor/375334958/sizes/m/#cc_license)

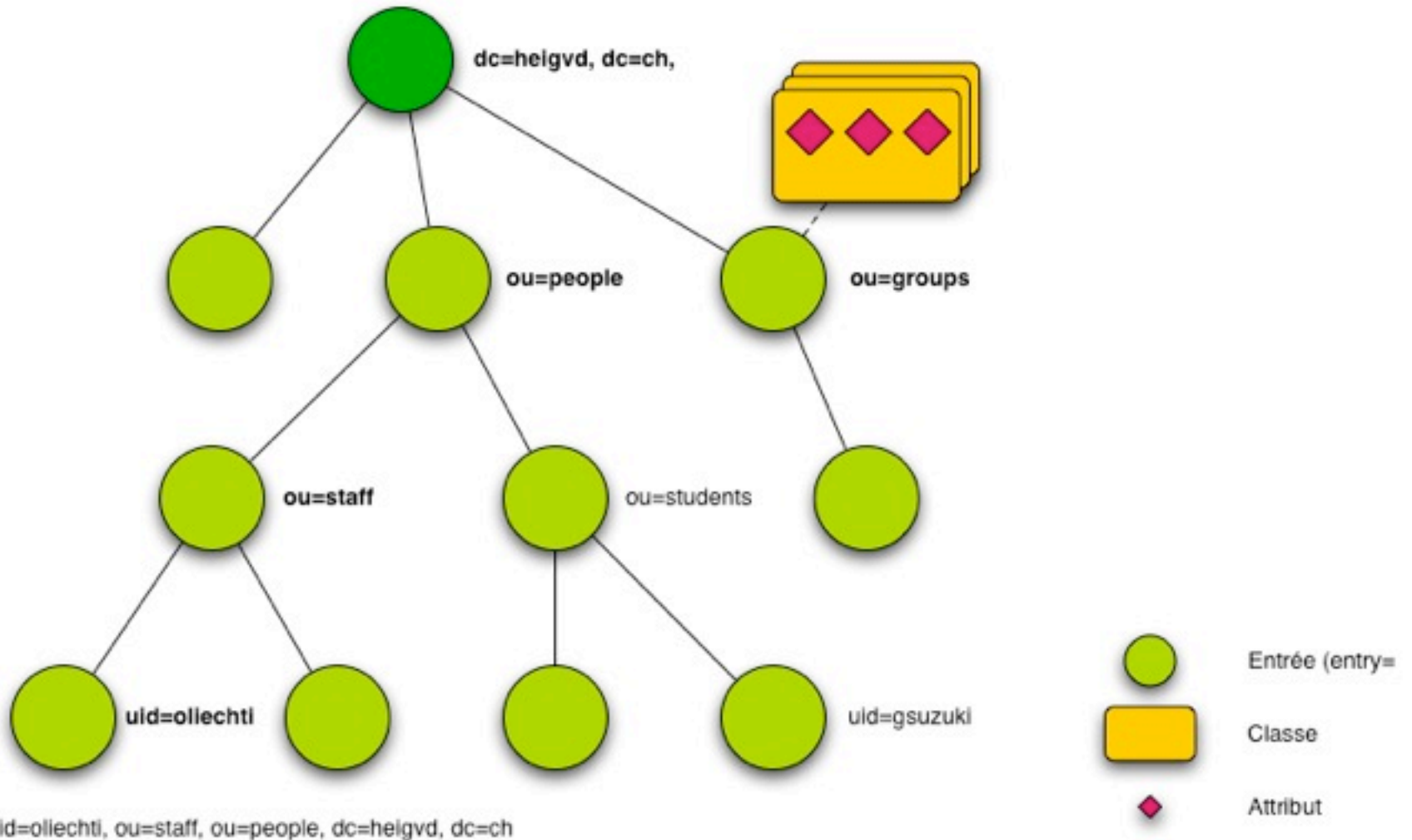
# LDAP: un annuaire pour partager des données



# *LDAP: le modèle de données*

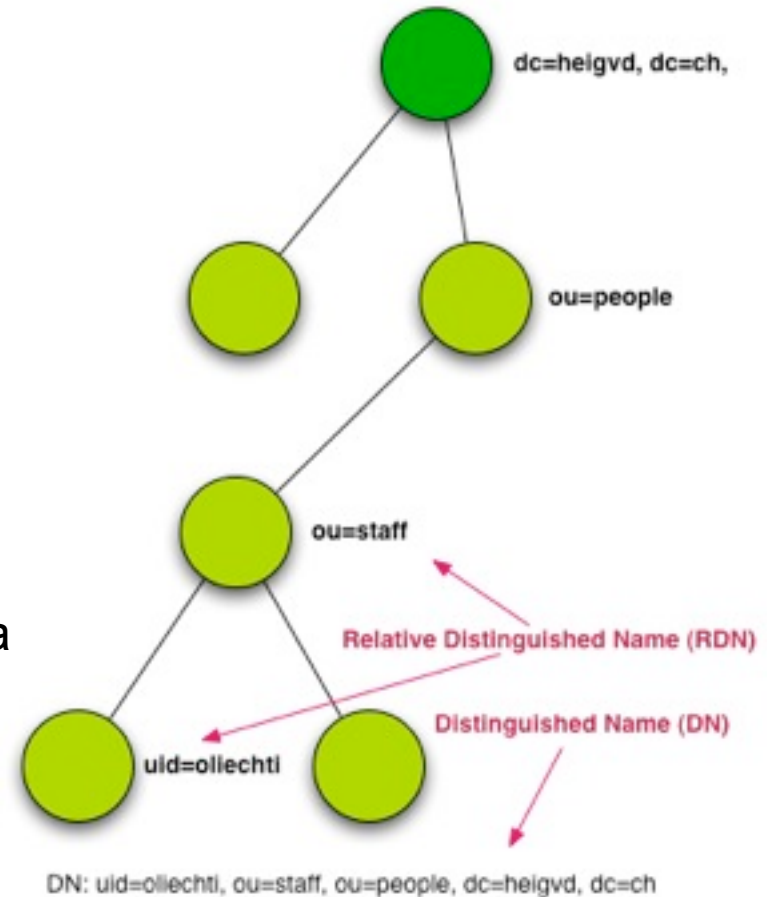


# LDAP: modèle de données



# Le Directory Information Tree (DIT)

- > **Les données sont organisées dans une structure hiérarchique.**
  - La racine est appelée "root", "suffix" ou encore "base".
  - Chaque noeud est appelé "entrée".
  - Les noeuds intermédiaires sont des "containers".
- > **Les entrées sont nommées:**
  - Le **Distinguished Name (DN)** permet de retrouver l'entrée dans la structure.
  - Le DN indique le "chemin" à suivre dans la structure pour retrouver l'entrée.
  - Le **Relative Distinguished Name (RDN)** identifie l'entrée de manière unique par rapport au parent.



# Comment définir la structure du DIT?

## > Que stocke-t-on dans l'annuaire?

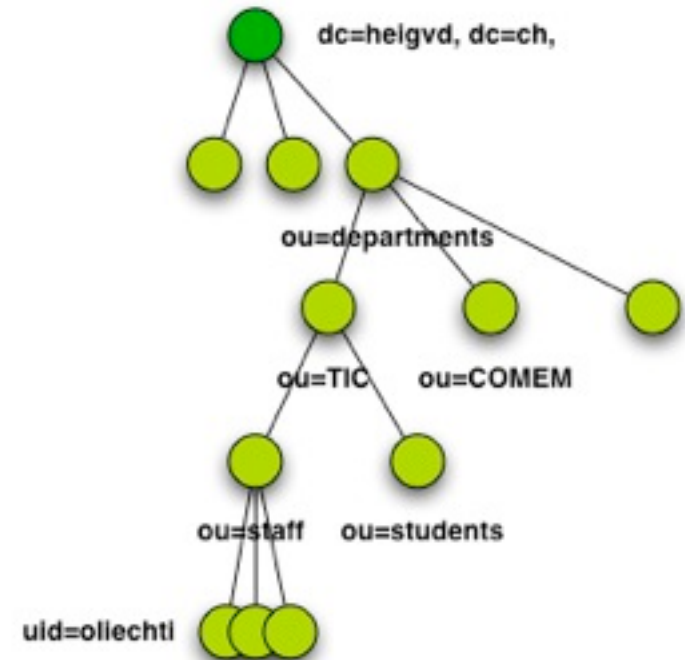
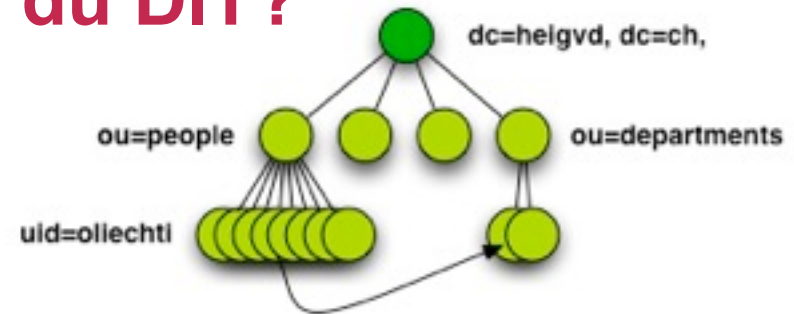
- Des personnes?
- Des équipements?
- Des services logiciels (objets)?
- Des paramètres de configuration?

## > Dans le cas de personnes, comment capturer la structure de l'organisation (entreprise, départements, groupes, etc.)

- Structuration par département?
- Structuration par pays?

## > Recommandation

- Privilégier une structure relativement plate, pour plus de flexibilité (appartenance à plusieurs départements, mutations, etc.)



# La notion d'entrée (entry)

- > Une "entrée" LDAP est simplement un objet qui est stocké dans l'annuaire.
- > C'est un "noeud" du DIT.
- > Une entrée est identifiée globalement de manière unique par son Distinguished Name (DN).
- > Une entrée est identifiée localement (par rapport à ses noeuds "frères") de manière unique par son Relative Distinguished Name (RDN).
- > L'état d'une entrée est défini par une liste d'attributs.
- > La structure d'une entrée (i.e. la liste des attributs) est définie dans une ou plusieurs "classe d'objet".
- > Quelques exemples d'entrées:
  - Une personne, un groupe, un département, une imprimante, un service en-ligne, un paramètre de configuration, etc.

# La notion de classe d'objet (object class)

- > La notion de "classe d'objet LDAP" est un peu similaire la notion de "classe" en programmation orientée objet.
- > Une classe d'objets spécifie une liste d'attributs
  - dont certains sont obligatoires
  - dont certains sont optionnels
- > Une classe d'objets peut étendre une autre classe d'objet.
- > La syntaxe utilisée pour définir une classe d'objet est définie dans le RFC 2252 (LDAPv3 Attribute Syntax Definitions)
- > Il existe de nombreuses classes d'objets standardisées, que l'on peut utiliser. Par exemple:
  - inetOrgPerson, OrganizationalPerson, Person
  - organizationalUnit
  - groupOfUniqueNames

# La notion de classe d'objet (object class)

```

ObjectClassDescription = "(" whsp
    numericoid whsp      ; ObjectClass identifier
    [ "NAME" qdescrs ]
    [ "DESC" qdstring ]
    [ "OBSOLETE" whsp ]
    [ "SUP" oids ]        ; Superior ObjectClasses
    [ ( "ABSTRACT" / "STRUCTURAL" / "AUXILIARY" ) whsp ]
                        ; default structural
    [ "MUST" oids ]       ; AttributeTypes
    [ "MAY" oids ]        ; AttributeTypes
whsp ")"

```

# Exemple: inetOrgPerson

> **Référence: /Users/oliechti/OpenDS/config/schema/00-core.ldif**

```
objectClasses: ( 2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson'  
  SUP organizationalPerson STRUCTURAL MAY ( audio $ businessCategory $  
  carLicense $ departmentNumber $ displayName $ employeeNumber $ employeeType $  
  givenName $ homePhone $ homePostalAddress $ initials $ jpegPhoto $  
  labeledURI $ mail $ manager $ mobile $ o $ pager $ photo $ roomNumber $  
  secretary $ uid $ userCertificate $ x500UniqueIdentifier $  
  preferredLanguage $ userSMIMECertificate $ userPKCS12 ) X-ORIGIN 'RFC 2798' )
```

# La notion d'attribut

- > Les attributs définissent l'état des entrées LDAP.
- > Les attributs sont définis dans des classes d'objets.
- > Chaque attribut a un type associé (String, Binary, etc.).
- > Les attributs peuvent être multi-valués.

```
attributeTypes: ( 0.9.2342.19200300.100.1.41  
  NAME ( 'mobile' 'mobileTelephoneNumber' ) EQUALITY telephoneNumberMatch  
  SUBSTR telephoneNumberSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.50  
  X-ORIGIN 'RFC 4524' )
```



# La notion de schéma

- > **Quand on déploie un service LDAP, on définit un "schéma" pour spécifier la structure des données qui peuvent être traitées par le service:**
  - Quelles sont les classes d'objets autorisées?
  - Quels sont les attributs autorisés?
- > **Il existe des schémas standardisés, qui décrivent des classes d'objets et des attributs standardisés.**
- > **Ces schémas sont disponibles après l'installation d'un serveur LDAP. Bien souvent, ils sont suffisants.**
- > **En cas de besoin, on peut étendre le schéma:**
  - avec des classes d'objets spécifiques (e.g. heigvdPerson)
  - avec des attributs spécifiques (e.g. gapsIdentifier)
- > **La manière dont on étend le schéma dépend de l'implémentation (OpenDS, OpenLDAP, etc.)**

# Object Identifier (OID)

- > **Un OID est une chaîne numérique qui est utilisée pour identifier certains éléments d'un annuaire.**
- > **Ils sont notamment utilisés pour identifier les classes d'objet et les attributs.**
- > **Quand on définit une nouvelle classe d'objet ou un nouvel attribut, on doit donc obtenir un OID:**
  - Si le schéma est utilisé uniquement en interne, on peut utiliser une base commune aux autres organisations (analogie: adressage IP privé).
  - Si le schéma est partagé avec l'extérieur, on doit obtenir un OID globalement unique.
- > **Les OIDs sont gérés par l'IANA; la procédure pour obtenir une plage est facile et rapide.**

<https://www.opensds.org/wiki/page/HowToExtendTheLDAPSchema#section-HowToExtendTheLDAPSchemaWorkingWithObjectIdentifiersOIDs>

LDAP object classes and attributes require a base object identifier (OID) that must be unique within your organization to avoid naming conflicts in the directory. If you plan to use your directory internally within your organization, use the OIDs provided in the OpenDS directory server. If you plan to export your schema or publicly expose your schema in any way, you should consider entering a request for a unique OID for your organization (see Obtaining a Base OID).

After you have obtained a base OID, you can add branches to it for your organization's object classes and attributes. For example, the OpenDS project uses an assigned base OID of 1.3.6.1.4.1.26027. For each component type, OpenDS provides unique branch numbers to the base OID for each schema component.

**Note:** OpenDS provides a [comprehensive set](#) of OIDs that should be sufficient for most applications. You can also [request OIDs](#) for addition to the OpenDS repository.

For example, OpenDS uses the following base OIDs for each schema component:

OID Value	Type
1.3.6.1.4.1.26027.1.1	Attribute
1.3.6.1.4.1.26027.1.2	Object classes
1.3.6.1.4.1.26027.1.3	Attribute syntaxes
1.3.6.1.4.1.26027.1.4	Matching rules
1.3.6.1.4.1.26027.1.5	Controls
1.3.6.1.4.1.26027.1.6	Extended operations
1.3.6.1.4.1.26027.1.9	General use (Currently, no OIDs are assigned for OpenDS.)
1.3.6.1.4.1.26027.1.999	Experimental use

Mozilla Firefox

http://www.iana.org/assignments/enterprise-numbers

oid 15103

Most Visited ▾ WebStamp Business ... Getting Started Latest Headlines ▾ Connectors for Dash... uBike MyData MyAccount Welcome Welcome >>

Contacts ▾ Events ▾ Locations ▾ Tagspaces ▾ Bookmarks ▾ Resources ▾ Options

Disable ▾ Cookies ▾ CSS ▾ Forms ▾ Images ▾ Information ▾ Miscellaneous ▾ Outline ▾ Resize ▾ Tools ▾ View Source ▾

http://www.ian...rprise-numbers x IP Numero d'affection OID aux ent... x

nantong vocational college  
guoping huang  
hgp@mail.ntvc.edu.cn

26020  
DePratti Consulting LLC  
Patrick DePratti  
pdepratti@yahoo.com

26021  
Ligos Corporation  
Jim Weller  
jweller@ligos.com

26022  
Kamayo  
Julien Nitard  
julien.nitard@m4tp.org

26023  
Fachschaft MPI, TU München  
Sebastian Hanigk  
shanigk@fs.tum.de

26024  
subnet - platform for media art and experimental technologies  
Andreas Förster  
andreas@subnet.at

26025  
Ari Voutilainen  
Ari Voutilainen  
ari.voutilainen@iki.fi

26026  
arm4.org  
David Carter  
dcarter@entertain-me.com

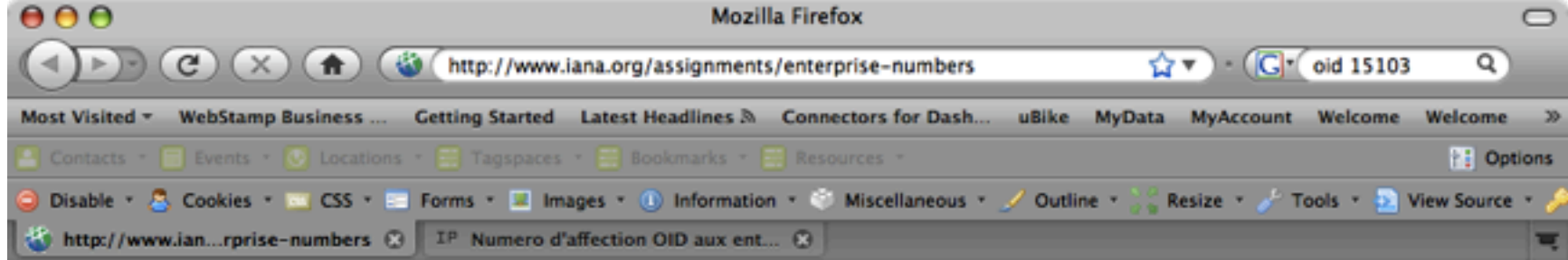
26027  
OpenDS.org  
OpenDS Administrator  
opens@dev.java.net

26028  
MetaSoft  
Ilya Melamed  
ilya77@gmail.com

26029  
DuroSystems Ltd.  
Brett Doyle

Find: 26027 Next Previous Highlight all Match case

Done



## PRIVATE ENTERPRISE NUMBERS

(last updated 2008-07-11)

SMI Network Management Private Enterprise Codes:

Prefix: iso.org.dod.internet.private.enterprise (1.3.6.1.4.1)

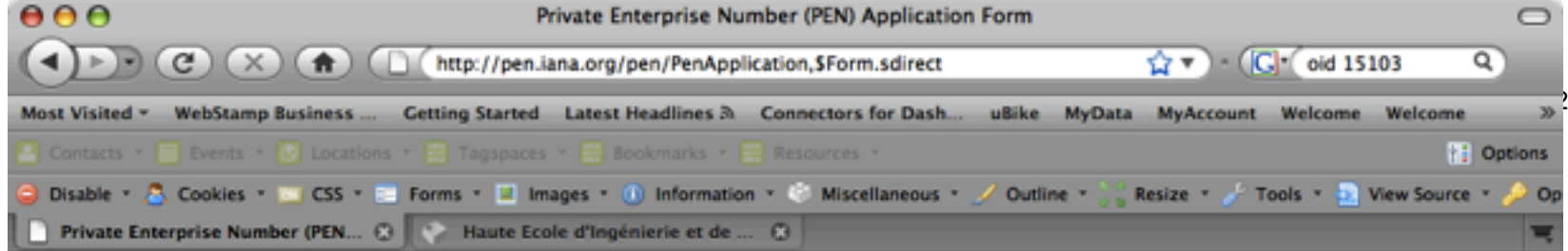
This file is <http://www.iana.org/assignments/enterprise-numbers>

### Decimal

	Organization	Contact	Email
0	Reserved		
	Internet Assigned Numbers Authority		iana@iana.org
1	NxNetworks	Michael Kellen	OID.Admin@NxNetworks.com
2	IBM	Bob Moore	remoore@us.ibm.com
3	Carnegie Mellon	Mark Poepping	host-master@andrew.cmu.edu
4	Unix	Keith Sklower	sklower@oakeffe.berkeley.edu
5	ACC	Art Berggreen	art@SALT.ACC.COM
6	TWG	John Lunn	

Find: Q 26027 Next Previous Highlight all Match case

Done



[Request Private Enterprise Number \(PEN\)](#) | [Modify Private Enterprise Number \(PEN\)](#) | [Enterprise Numbers](#) | [Contact IANA](#) | [IANA](#)

## Application Information Confirmation

Please verify that the information you have provided is correct and click the "Confirm" button to submit the application for IANA review. If you would like to make corrections to the application you are submitting, click "Make Changes". Click "Cancel" to exit without submitting the information to IANA.

### Organization

**Organization Name:** Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud (HEIG-Vd)  
**Organization Address:** Av. des Sports 20  
**Organization Phone:** +41 24 55 77584

### Contact

**Contact Name:** Olivier Liechli  
**Contact Address:** Av. des Sports 20  
**Contact Phone:** +41 24 55 77584  
**Contact Fax:**  
**Contact Email:** olivier.liechli@heig-vd.ch

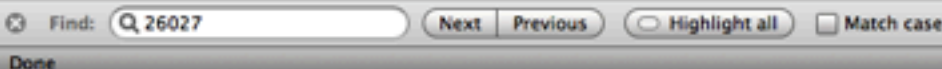
Confirm

Make Changes

Cancel

<http://pen.iana.org/pen/PenApplication.page>

heig



.ch

## Comment gérer "ses" OIDs?

1.3.6.1.4.1.xxx.n.n.n

Prefix: iso.org.dod.internet.private.enterprise (1.3.6.1.4.1)

Numéro assigné à l'IANA à la HEIG-Vd

Structure d'identification gérée librement par la HEIG-Vd

.1.\*: test

.2.\*: teaching .2.1.\*: PDA

.3.\*: research

.4.\*: prod

# *LDAP: le protocole*



# LDAP: le protocole

- > **LDAP est un protocole client-serveur:**
  - qui utilise TCP
  - dont le port standard est 389
- > **Principale commandes LDAP:**
  - **Bind** (authentification et établissement d'une session)
  - **Search** - search for and/or retrieve directory entries
  - **Add** a new entry
  - **Delete** an entry
  - **Modify** an entry
  - **Modify** Distinguished Name (DN) - move or rename an entry
  - **Unbind** (fin de la session)

# Messages

- > **La structure des messages est spécifiée au moyen de la notation ASN.1**
  - Abstract Standard Notation One (standard ISO/IUT)
  - ASN est une notation abstraite pour décrire des structures de données associées aux protocoles de communication
  - Il existe différents moyens d'encoder une structure décrite avec ASN.1.
  - LDAP utilise un sous-ensemble du format d'encodage Basic Encoding Rule (BER)
- > **Il n'est donc pas facile de tester LDAP dans une session telnet (contrairement à HTTP par exemple)**

# LDAP & ASN.1: exemple d'implémentation

```

LDAPMessage ::= SEQUENCE {
    messageID      MessageID,
    protocolOp     CHOICE {
        bindRequest      BindRequest,
        bindResponse     BindResponse,
        unbindRequest    UnbindRequest,
        searchRequest     SearchRequest,
        searchResEntry    SearchResultEntry,
        searchResDone     SearchResultDone,
        searchResRef      SearchResultReference,
        modifyRequest     ModifyRequest,
        modifyResponse    ModifyResponse,
        addRequest        AddRequest,
        addResponse       AddResponse,
        delRequest        DelRequest,
        delResponse       DelResponse,
        modDNRequest      ModifyDNRequest,
        modDNResponse     ModifyDNResponse,
        compareRequest    CompareRequest,
        compareResponse   CompareResponse,
        abandonRequest    AbandonRequest,
        extendedReq       ExtendedRequest,
        extendedResp      ExtendedResponse },
    controls        [0] Controls OPTIONAL }

```

<http://directory.apache.org/apacheds/1.0/asn1-codec.html>

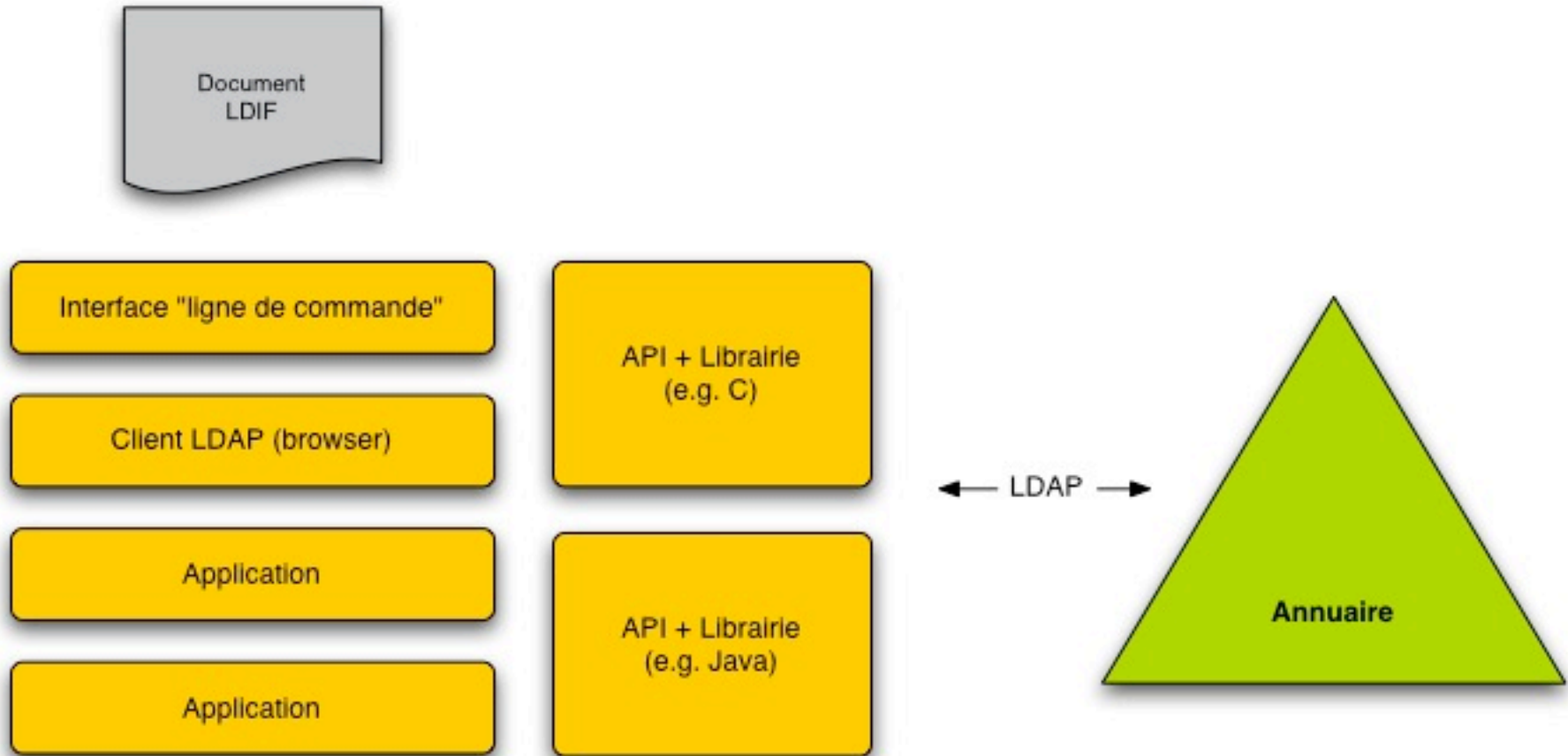
<http://directory.apache.org/apacheds/1.0/ldap-asn1-codec.html>



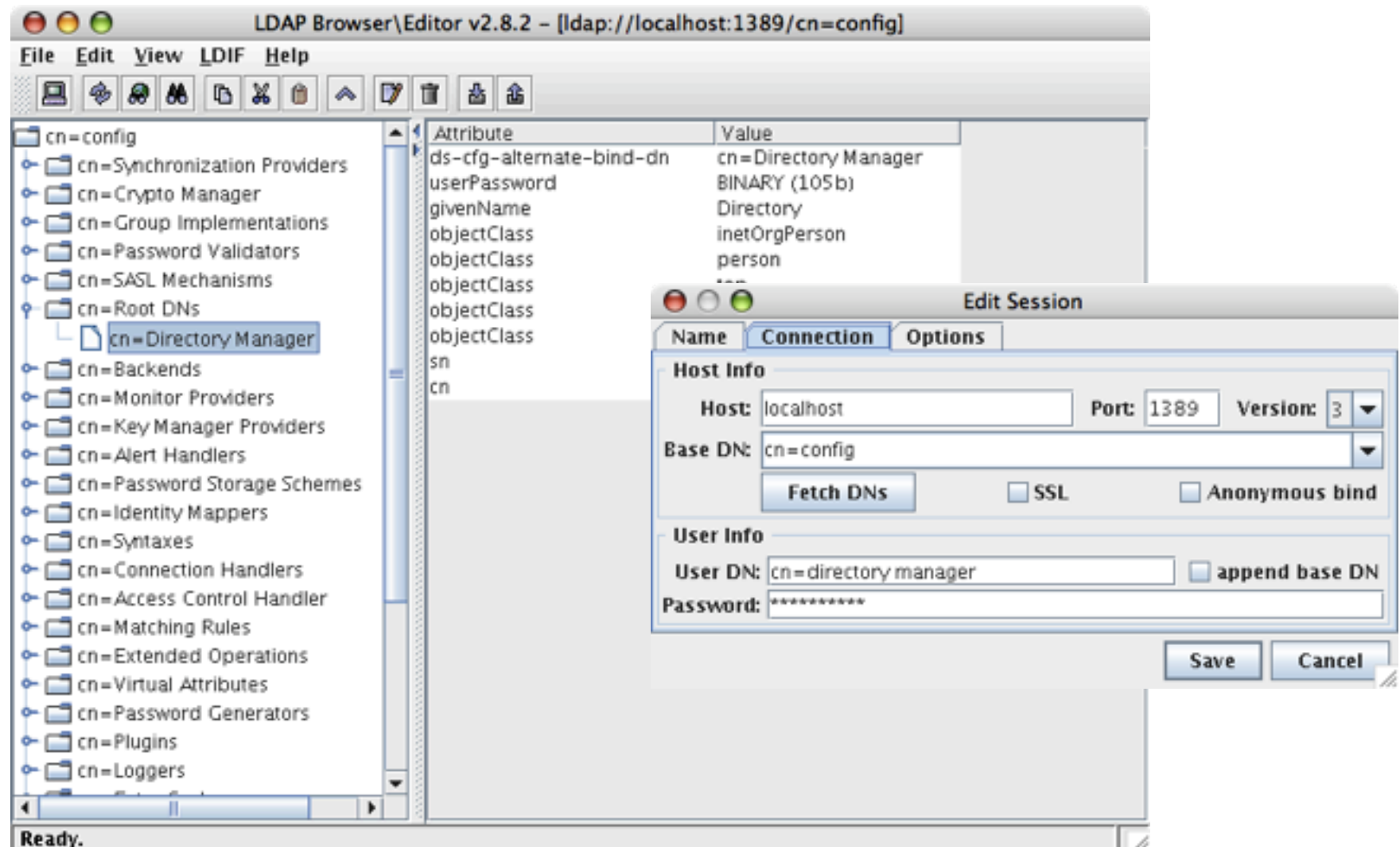
# *LDAP: l'infrastructure*

# LDAP: composants

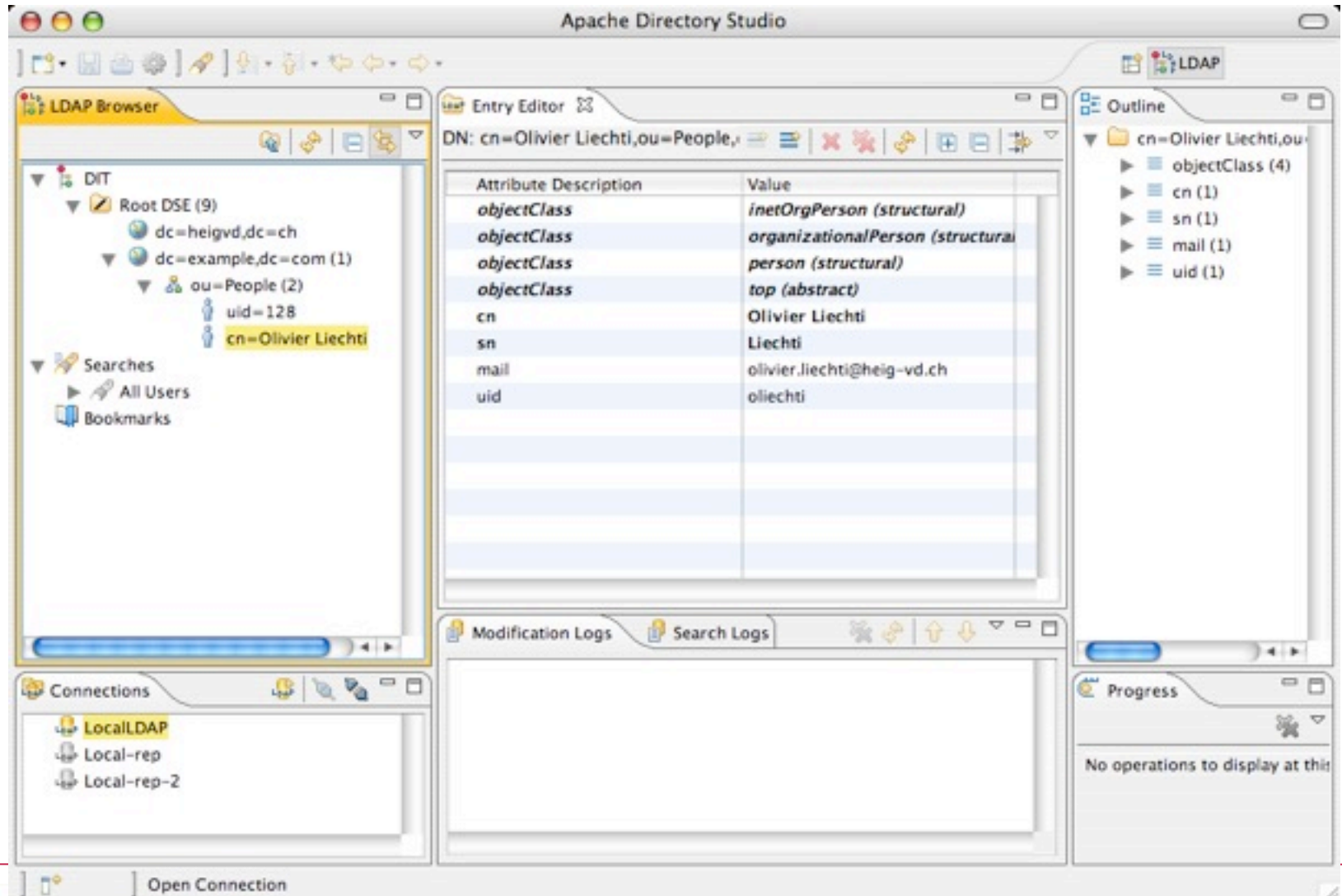
<http://www.ietf.org/rfc/rfc2849.txt>



# LDAPBrowser



# Apache Directory Studio



# *OpenDS*



# OpenDS

- > **OpenDS est un annuaire LDAP:**
  - développé en Open Source, avec le support de Sun Microsystems
  - 100% Java
  - qu'on peut intégrer dans d'autres applications ("embeddable")
  - qu'on peut étendre de manière flexible
- > **L'installation d'OpenDS est très simple:**
  - installation via Java WebStart
  - structure de fichier simple
- > **En même temps, OpenDS intègre des fonctions "entreprise"**
  - réplication
  - performances
- > **OpenDS permet ainsi une expérimentation rapide et aisée!**
- > **<http://www.opensds.org/>**



# OpenDJ

- > **OpenDJ est un fork de OpenDS:**
  - développé en Open Source, par la société ForgeRock
  - 100% Java
- > **<http://www.forgerock.org/openssl.html>**



# Attention!

- > **Beaucoup de systèmes d'exploitation (Mac OS, Solaris, Linux, etc.) intègrent des commandes ldap de base:**
  - Exemple sur Mac OS: `/usr/bin/ldapsearch`
  - Ces commandes sont souvent dans le chemin d'accès (PATH)
- > **Les serveurs LDAP (p.ex. OpenDS) fournissent leurs propres commandes, avec des différences dans la syntaxe et les options:**
  - Exemple: `${OPEN_DS_INSTALL_PATH}/bin/ldapsearch`
  - Ces commandes ne sont généralement pas dans le chemin d'accès
- > **Quand vous faites vos tests:**
  - Faites bien attention à la commande que vous utilisez:
    - `mv ${OPEN_DS_INSTALL_PATH}/bin/ldapsearch`
    - `./ldapsearch`
  - Est différent de:
    - `mv ${OPEN_DS_INSTALL_PATH}/bin/ldapsearch`
    - `ldapsearch`

# ldapsearch (1)

> **Commande utilisée pour extraire des données de l'annuaire**

> **Syntaxe:**

– ldapsearch [options] [filter] [attributes]

> **Options importantes:**

- -h, --host à quel serveur veut-on se connecter?
- -p, --port sur quel port écoute-t-il?
- -D, --bindDN avec quel identité veut-on se connecter?
- -w, --bindPassword avec quel mot de passe (à éviter, penser à `ps`!!)
- -b, --baseDN à partir d'où veut-on faire la recherche?
- -a, --searchScope avec quelle profondeur?
- -T, --dontWrap pour éviter les ruptures de lignes (LDIF)
- --propertiesFilePath pour éviter de saisir toutes les options

> **Référence:**

– <https://www.openss.org/wiki/page/Ldapsearch>

## Idapsearch (2)

### > Syntaxe du filtre

- Définie dans le RFC 2254
- Opérateurs pour filtres complexes: &, |, !

### > Quelques exemples

- Entrées dont l'attribut cn est égal à Babs Jensen:  
→ (cn=Babs Jensen)
- Entrées dont l'attribut cn est différent de Tim Howes:  
→ (!(cn=Tim Howes))
- Personnes dont le nom de famille est Jensen ou dont le prénom est Babs et le nom de famille commence par J:  
→ (&(objectClass=Person)(|(sn=Jensen)(cn=Babs J\*))

# ldapsearch (3)

## > Retourner toutes les entrées

- `ldapsearch -h hostname -p 389 -b dc=example,dc=com "(objectclass=*)"`

```
dn: dc=example,dc=com
objectClass: domain
objectClass: top
dc: example
```

```
dn: ou=Groups,dc=example,dc=com
objectClass: organizationalunit
objectClass: top
ou: Groups
```

```
dn: cn=Directory
Administrators,ou=Groups,dc=example,dc=com
objectClass: groupofuniqueNames
objectClass: top
ou: Groups
cn: Directory Administrators
uniqueMember: uid=kvaughan, ou=People,
dc=example,dc=com
uniqueMember: uid=rdaugherty, ou=People,
dc=example,dc=com
uniqueMember: uid=hmiller, ou=People, dc=example,dc=com
```

## > Retourner certains attributs

- `ldapsearch -h hostname -p 389 -b dc=example,dc=com "(cn=Sam Carter)" telephoneNumber`

```
dn: uid=scarter,ou=People,dc=example,dc=com
telephonenumber: +1 408 555 4798
```

# ldapmodify (1)

- > **Commande utilisée pour ajouter des données de l'annuaire**
- > **Syntaxe:**
  - `ldapmodify [options] [filter] [attributes]`
- > **Options importantes:**
  - `-h, --host` à quel serveur veut-on se connecter?
  - `-p, --port` sur quel port écoute-t-il?
  - `-D, --bindDN` avec quel identité veut-on se connecter?
  - `-w, --bindPassword` avec quel mot de passe (à éviter, penser à `ps`!!)
  - `-f, --filename` le fichier où se trouvent les données LDIF
- > **Deux possibilités pour fournir les informations au serveur:**
  - saisir du LDIF en ligne de commande + CTRL-D (\*nix) ou CTRL-Z (Win)
  - utiliser l'option `-f` et saisir le LDIF dans un fichier
- > **Référence:**
  - <https://www.openss.org/wiki/page/Ldapmodify>

## Exemple LDIF pour ajouter une entrée

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: add
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
uid: john.doe
givenName: John
sn: Doe
cn: John Doe
mail: john.doe@example.com
userPassword: password
```



## Exemple LDIF pour modifier une entrée

```
dn: uid=john.doe,ou=People,dc=example,dc=com
changetype: modify
replace: description
description: This is the new description for John Doe
-
add: mailAlternateAddress
mailAlternateAddress: jdoe@example.com
```

# Exemple LDIF pour supprimer une entrée

```
dn: uid=john.doe,ou=People,dc=example,dc=com  
changetype: delete
```

# La commande dsconfig

>>>> OpenDS configuration console main menu

What do you want to configure?

- |  |                              |
|--|------------------------------|
| 1) Access Control Handler              | 20) Log Rotation Policy      |
| 2) Account Status Notification Handler | 21) Matching Rule            |
| 3) Alert Handler                       | 22) Monitor Provider         |
| 4) Attribute Syntax                    | 23) Password Generator       |
| 5) Backend                             | 24) Password Policy          |
| 6) Certificate Mapper                  | 25) Password Storage Scheme  |
| 7) Connection Handler                  | 26) Password Validator       |
| 8) Crypto Manager                      | 27) Plugin                   |
| 9) Debug Target                        | 28) Plugin Root              |
| 10) Entry Cache                        | 29) Replication Domain       |
| 11) Extended Operation Handler         | 30) Replication Server       |
| 12) Global Configuration               | 31) Root DN                  |
| 13) Group Implementation               | 32) Root DSE Backend         |
| 14) Identity Mapper                    | 33) SASL Mechanism Handler   |
| 15) Key Manager Provider               | 34) Synchronization Provider |
| 16) Local DB Index                     | 35) Trust Manager Provider   |
| 17) Local DB VLV Index                 | 36) Virtual Attribute        |
| 18) Log Publisher                      | 37) Work Queue               |
| 19) Log Retention Policy               |                              |
| q) quit                                |                              |

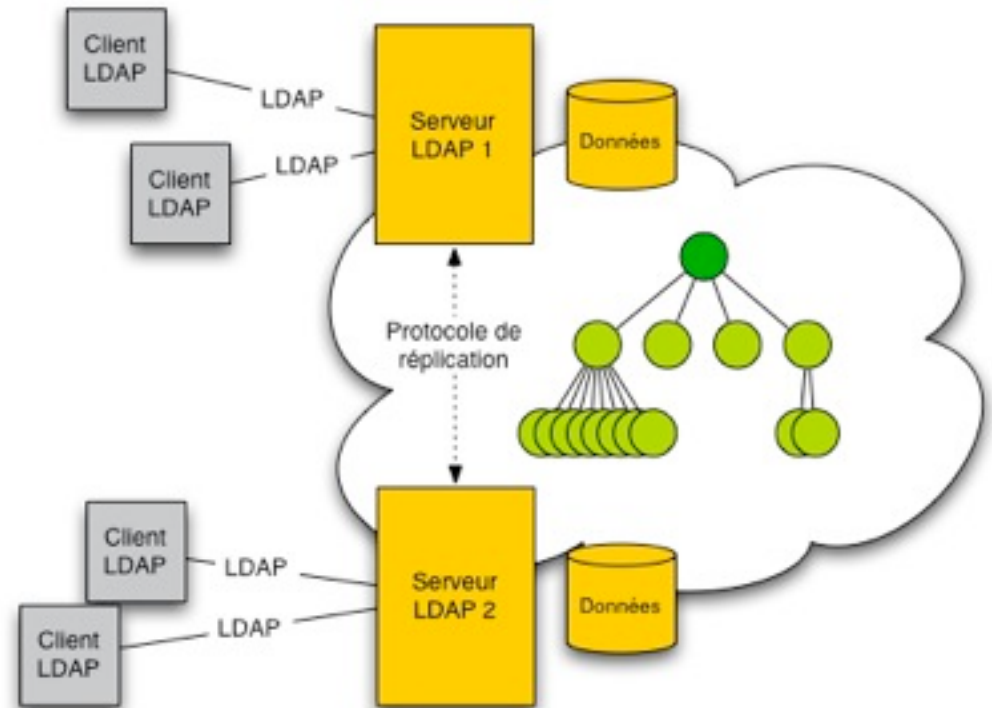
Enter choice: 35

# *Réplication*

# Principes de la réplication LDAP

## > Qu'entend-on par "réplication"?

- Plusieurs serveurs LDAP sont déployés:
  - ➔ soit dans le même centre de calcul;
  - ➔ soit dans des centres de calcul différents (p.ex. différents pays)
- Les données accessibles du service d'annuaire sont copiées (répliquées) entre ces serveurs.
- Les clients accèdent à l'un ou l'autre de ces serveurs (en fonction des objectifs et de la topologie).



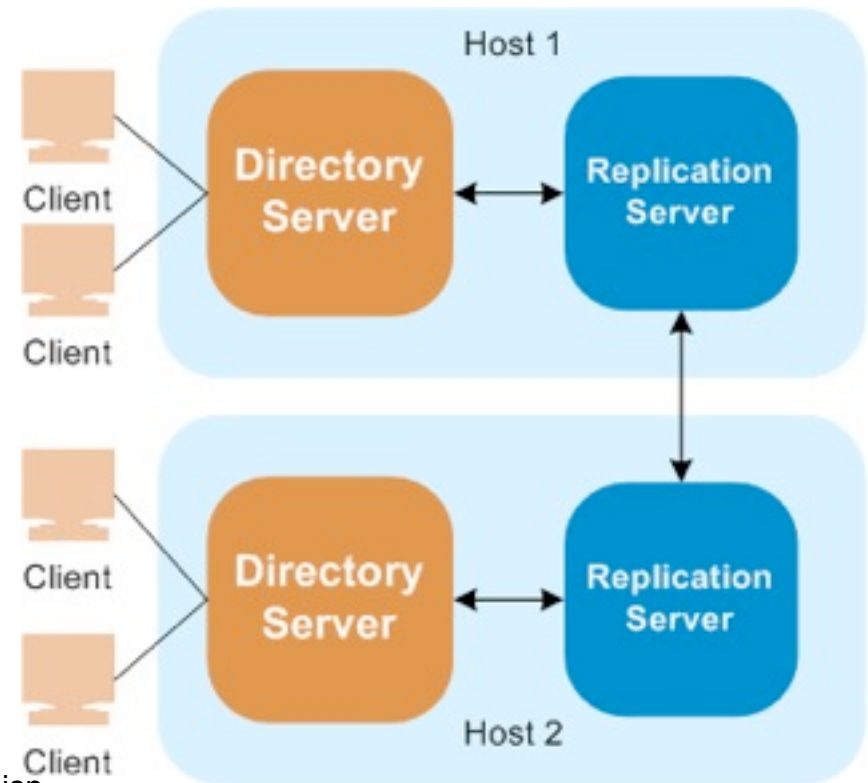
# Principes de la réplication LDAP

## > Pourquoi utiliser la réplication?

- Pour assurer les qualités systémiques du service LDAP!
- Performance
- Scalabilité
- Disponibilité

## > Différents types de topologies:

- Single Master (1 server accepte les écritures)
- Multi Master (plusieurs serveurs acceptent les écritures)



[http://www.sun.com/bigadmin/features/articles/dsee6\\_multimaster.jsp](http://www.sun.com/bigadmin/features/articles/dsee6_multimaster.jsp)

<https://www.opens.org/wiki/page/SmallTopologies>

OpenDS Status Panel

### Server Status

Server Run Status: Started

Open Connections: 2

### Server Details

Host Name: olivier-liechti-computer.local  
 Administrative Users: cn=Directory Manager  
 Installation Path: /Users/oliehti/OpenDS-rep-2  
 OpenDS Version: OpenDS Directory Server 1.1.0-build001  
 Java Version: 1.5.0\_07

### Connection Handlers

Address:Port	Protocol	Status
0.0.0.0:161	SNMP	Disabled
0.0.0.0:1689	JMX	Disabled
0.0.0.0:3389	LDAP	Enabled
0.0.0.0:636	LDAPS	Disabled

### Data Sources

Base DN	Backend ID	Entries	Replication	Missing Changes	Age of Oldest Missing Change
dc=example,dc=com	userRoot	2002	Enabled	0	<not available>

OpenDS

OpenDS Status Panel

### Server Status

Server Run Status: Started

Open Connections: 3

### Server Details

Host Name: olivier-liechti-computer.local  
 Administrative Users: cn=Directory Manager  
 Installation Path: /Users/oliehti/OpenDS-rep  
 OpenDS Version: OpenDS Directory Server 1.1.0-build001  
 Java Version: 1.5.0\_07

### Connection Handlers

Address:Port	Protocol	State
0.0.0.0:161	SNMP	Disabled
0.0.0.0:1689	JMX	Disabled
0.0.0.0:2389	LDAP	Enabled
0.0.0.0:636	LDAPS	Disabled

### Data Sources

Base DN	Backend ID	Entries	Replication	Missing Changes	Age of Oldest Missing Change
dc=example,dc=com	userRoot	2002	Enabled	0	<not available>

OpenDS

# Support de la réplication dans OpenDS

```
$ lsof -P -i TCP | grep 89 | grep LISTEN
```

```
java      6145 oliehti   33u  IPv6 0x7a4c46c  
java      9527 oliehti   40u  IPv6 0x798da24  
java      9527 oliehti   46u  IPv6 0x79beaf0  
java      9617 oliehti   41u  IPv6 0x7a2f174  
java      9617 oliehti   46u  IPv6 0x7a2dde8
```

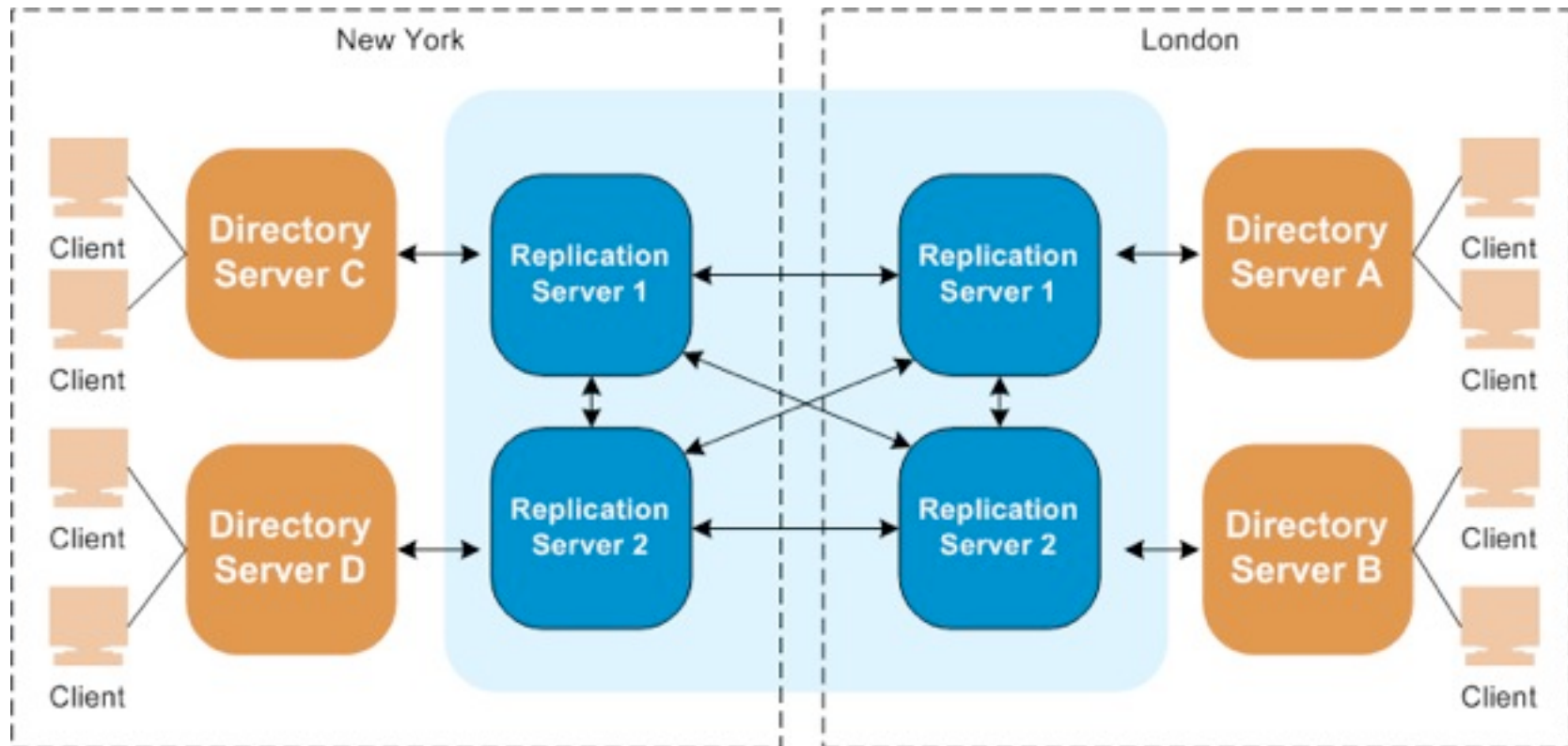
```
0t0  TCP *:1389 (LISTEN)  
0t0  TCP *:2389 (LISTEN)  
0t0  TCP *:8989 (LISTEN)  
0t0  TCP *:3389 (LISTEN)  
0t0  TCP *:9989 (LISTEN)
```



replication ports



# Topologie multi-site

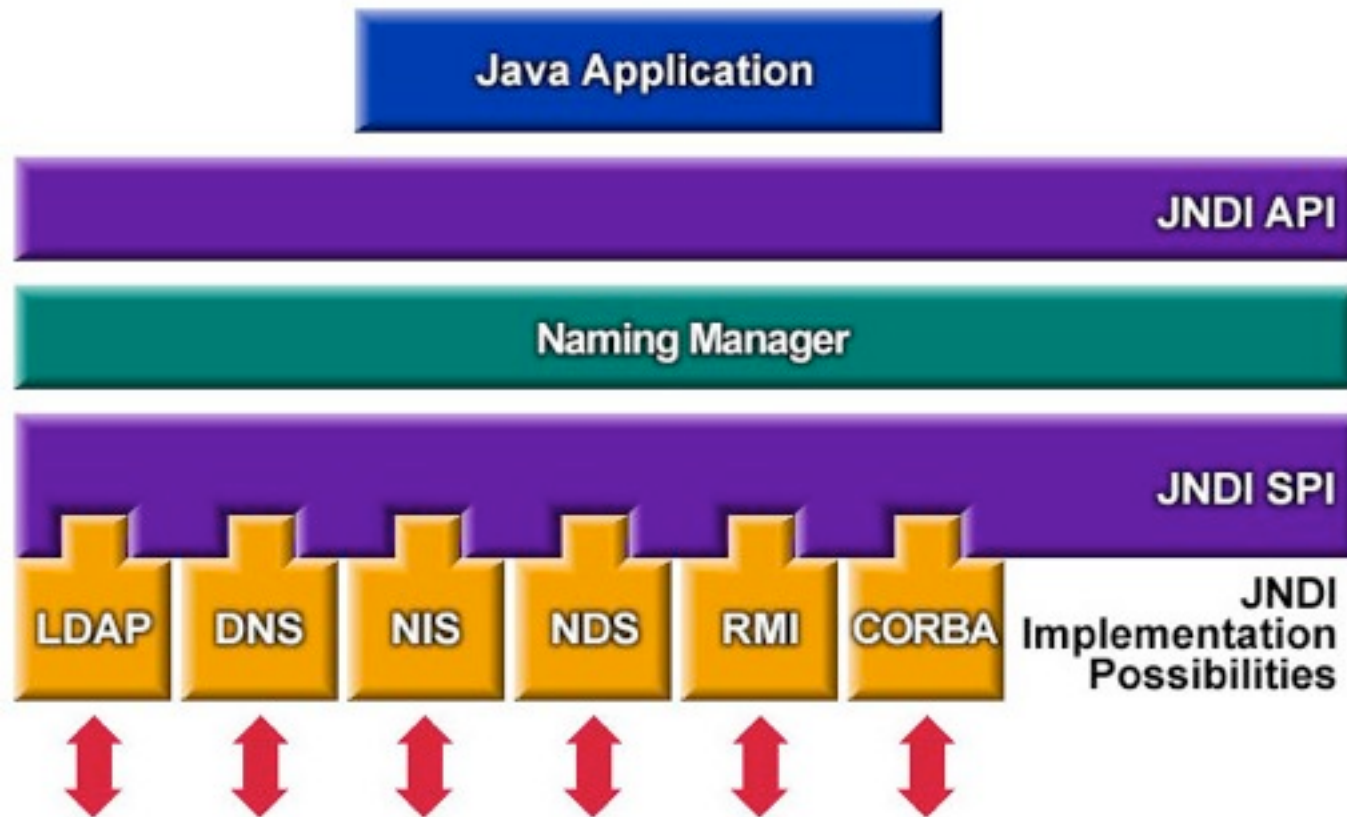


# *LDAP avec Java* *Java Naming & Directory* *Interface (JNDI)*

# Java Naming and Directory Interface (JNDI)

- > **LDAP est un des protocoles qui permet d'accéder à une structure de données hiérarchique.**
- > **Les développeurs Java aimeraient:**
  - avoir une API leur permettant d'accéder facilement à un service de gestion de données hiérarchique;
  - pouvoir utiliser cette API pour accéder à n'importe quel serveur LDAP (Active Directory, OpenDS, OpenLDAP, etc.).
- > **En d'autres termes, ils aimeraient avoir l'équivalent de l'API "JDBC" dans un contexte LDAP.**
- > **JNDI est la réponse à ces besoin. L'API offre un interface standardisée pour interagir avec des services de nommage et d'annuaire.**

# Java Naming and Directory Interface (JNDI)



<http://java.sun.com/products/jndi/tutorial/getStarted/overview/index.html>

# Comment utiliser JNDI?

- > La première étape consiste à "configurer" la connexion vers un service d'annuaire.
- > Cela se fait grâce à la classe `InitialDirContext`:

```
// Set up the environment for creating the initial context  
Hashtable env = new Hashtable();
```

```
env.put(Context.INITIAL_CONTEXT_FACTORY,  
        "com.sun.jndi.ldap.LdapCtxFactory");
```

```
env.put(Context.PROVIDER_URL,  
        "ldap://localhost:389/o=JNDITutorial");
```

```
DirContext ctx = new InitialDirContext(env);
```

# Comment utiliser JNDI?

- > **On peut ensuite faire des recherches, avec différentes méthodes à disposition. On peut par exemple utiliser les "filtres" LDAP**

```
// Create the default search controls
SearchControls ctls = new SearchControls();

// Specify the search filter to match
// Ask for objects that have the attribute "sn" == "Geisel"
// and the "mail" attribute
String filter = "(&(sn=Geisel)(mail=*))";

// Search for objects using the filter
NamingEnumeration answer = ctx.search("ou=People", filter, ctls);
```

# Comment utiliser JNDI?

- > **Quand on a une référence vers un objet LDAP, on peut parcourir ses attributs. Attention: chaque attribut peut avoir plusieurs valeurs!**

```
// Search for objects using the filter
NamingEnumeration answer = ctx.search("ou=People", filter, ctls);

for (NamingEnumeration ae = answer.getAll(); ae.hasMore();) {
    Attribute attr = (Attribute)ae.next();
    System.out.println("attribute: " + attr.getID());

    /* Print each value */
    for (NamingEnumeration e = attr.getAll(); e.hasMore();
        System.out.println("value: " + e.next()));
}
```