

# LDAP

Julien Baudry  
Ingénieur de Recherche  
LITIS – Le Havre

# Présentation et objectifs du cours

- Ce cours a pour objectif de vous présenter :
  - les annuaires LDAP
  - une implémentation libre proposant à la fois un serveur LDAP mais un ensemble d'outils clients : le projet OpenLDAP.
- Vous allez découvrir à travers ce cours les notions liées aux annuaires LDAP.
- Vous apprendrez également
  - Comment mettre en oeuvre un serveur OpenLDAP
  - Utiliser les commandes clientes fournies par le projet.
- Un dernier chapitre vous proposera un exercice concret : la connexion d'une application à l'annuaire mis en place.

# Organisation du travail

- Le cours est divisé en trois parties majeures :
  - une partie théorique concernant les annuaires LDAP
  - une partie pratique présentant le projet OpenLDAP
  - une partie pratique présentant comment connecter une application à OpenLDAP
    - Samba, Comptes Unix, Application web ...

# Pré-requis

- Maîtriser le shell et les commandes systèmes GNU/linux de base
- Maîtriser la gestion des droits Unix
- Maîtriser un langage de script (php, perl ...)

# Plan

- **Introduction**
- Protocole
- Modèle d'information
- Modèle de nommage
- Modèle fonctionnel
- Modèle de sécurité
- Modèle de duplication
- Conception d'un annuaire
- Architecture
- Configuration et administration d'un serveur OpenLdap

# Introduction (1)

- Qu'est ce qu'un annuaire
  - Exemple : annuaire téléphonique
    - Cet annuaire regroupe différentes entrées contenant chacune des informations particulières :
      - Nom, prénom, numéro de téléphone et adresse.
      - Informations sont classées par département, puis par ville, puis enfin par nom.
  - Voici les caractéristiques communes aux annuaires :
    - Un annuaire présente un **ensemble défini de données**
      - (annuaire : nom, prénom, numéro de téléphone, adresse)
    - Il **organise** ces données
      - (annuaire : classées par département, villes, nom)
    - Il offre un service de **consultation**
      - (annuaire : diffusion au format papier)
    - Il peut **protéger** les données
      - (annuaire : liste rouge)
    - Il est **plus consulté** que mis à jour
    - Il est **disponible** de manière permanente

# Introduction (2)

- Qu'est-ce que LDAP ?
  - Lightweight Directory Access Protocol.
  - normaliser l'interface d'accès aux annuaires.
  - faciliter le partage et la gestion des informations.
- Que peut m'apporter LDAP ?
  - simplifier la gestion des profils de personnes et de ressources.
    - Gains économiques + meilleure qualité
  - favoriser l'interopérabilité des systèmes d'information.
    - Gains économiques (réutilisation de l'existant)

# Introduction (3)

- Pourquoi LDAP est-il aussi populaire ?
  - intégré dans les outils de l'ensemble des acteurs du marché.
    - Microsoft, Sun, IBM, Novell, Linux, etc.
  - standard retenu pour la gestion de la sécurité :
    - authentification forte
    - gestion des autorisations d'accès à des applications



# Introduction (4)

Quelles sont les différences entre LDAP et une base de données ?

## **Les bases de données :**

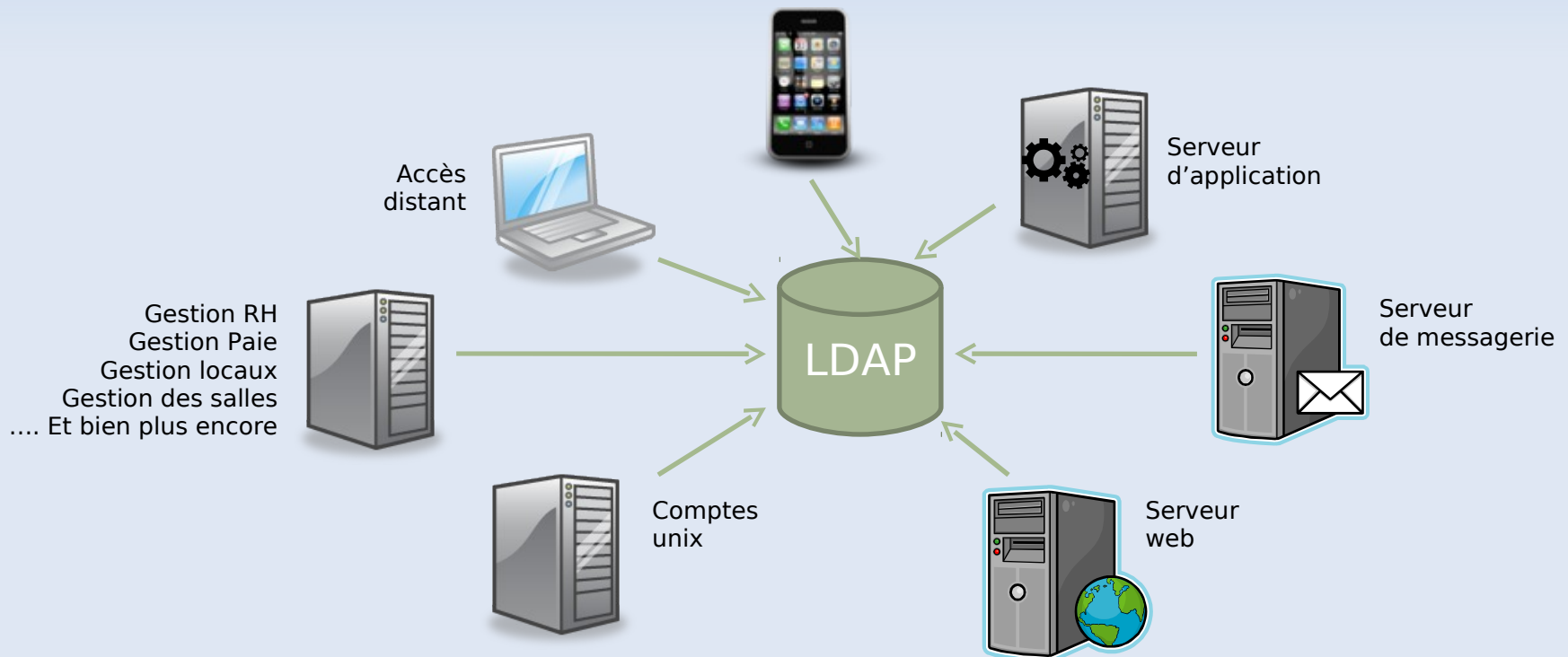
- stockage sous forme de tables.
- données stockées à un endroit unique.
- nommage spécifique à chaque application.
- critère de performance : transactions.

## **Les annuaires :**

- stockage hiérarchique.
- données globales et distribuées.
- nommage global et standardisé.
- critère de performance : accès en lecture.

# Introduction (5)

- Quels sont les objectifs ?
  - mettre en place un référentiel unique d'authentification.
    - exemple : authentification unifiée Unix/Windows.
  - fédérer plusieurs sources d'authentification.



# Introduction (6)

- Un bon exemple d'utilisation : Google
  - 1 compte -> n services
    - Mail, Docs, RSS Reader ...



# Introduction (7)

- Un peu d'histoire ...
  - 1988, L'UIT met au point les annuaires X.500 (protocole DAP)
    - uniformiser l'accès aux services
    - centraliser les ressources et les protéger
  - Problème :
    - DAP est compliqué à mettre en oeuvre
    - ne fonctionne pas sur les réseaux TCP/IP
  - 1993, l'Université du Michigan met au point le protocole LDAP
  - 1995, LDAP devient un protocole natif et utilisable indépendamment de X.500
  - Normalisé par l'IETF.
  - LDAP version 3 depuis 1997.
    - Unicode, TLS, SASL, Referrals ...

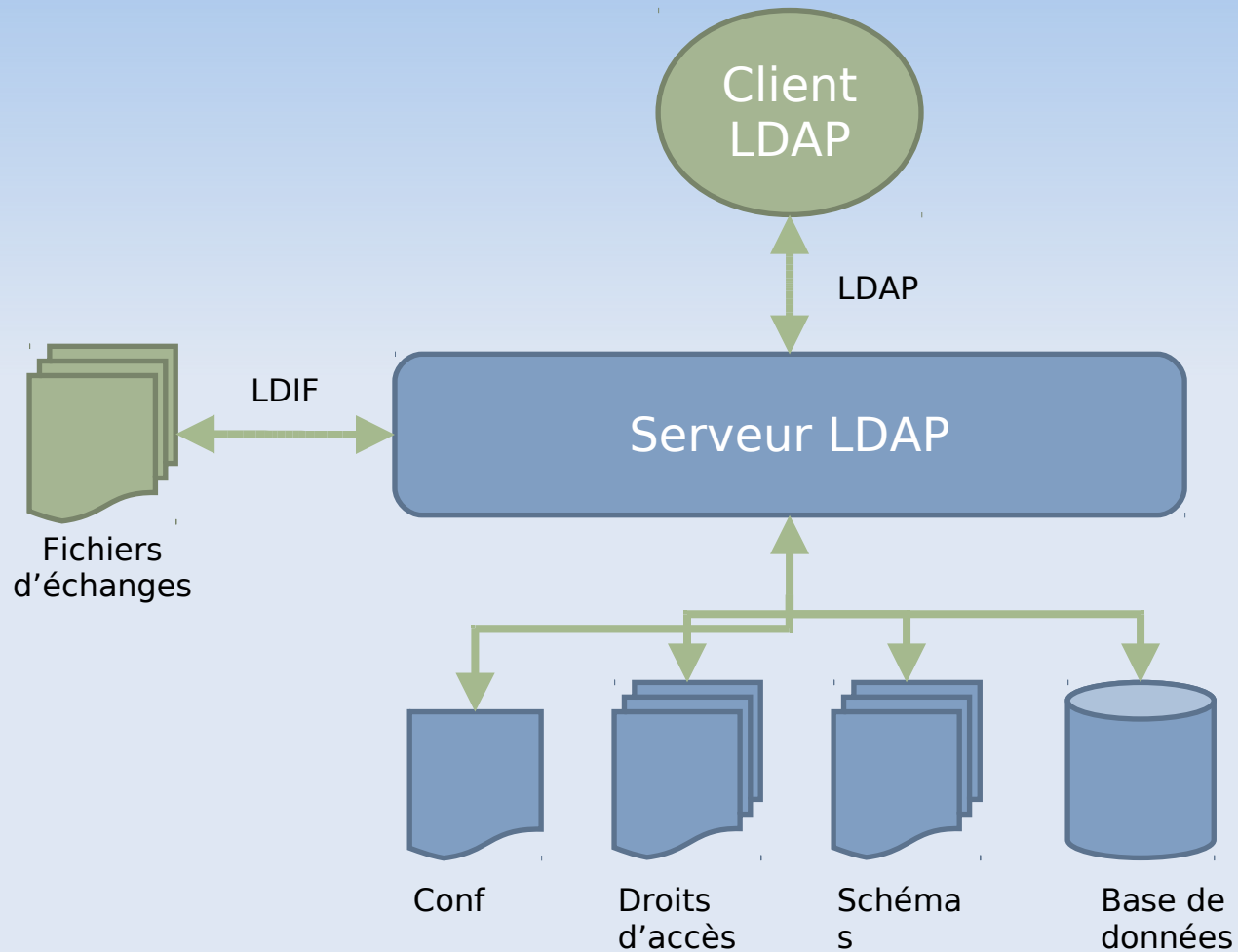
# Introduction (8)

- Quelques annuaires LDAP
  - Voici une liste des principaux annuaires LDAP existant sur le marché :
    - OpenLDAP : <http://www.openldap.org>
    - Apache Directory Server : <http://directory.apache.org>
    - Sun (One/Java) Directory Server :  
<http://www.sun.com>
    - Active Directory : <http://www.microsoft.com>
    - [...]

# Concepts LDAP

- Le standard LDAP
  - Protocole client-serveur basé sur TCP/IP.
  - Quatre modèles prédéfinis:
    - le modèle d'information (nature des données).
    - le modèle de nommage (structure hiérarchique).
    - le modèle des services (fonctions disponibles).
    - le modèle de sécurité (droits d'accès).
  - Des classes d'objets et des attributs normalisés.
- Des fonctions de recherche évoluées.
- Répartition des données sur plusieurs référentiels de manière transparente.

# Les composants d'un annuaire LDAP



# Plan

- Introduction
- **Protocole**
- Modèle d'information
- Modèle de nommage
- Modèle fonctionnel
- Modèle de sécurité
- Modèle de duplication
- Conception d'un annuaire
- Architecture
- Configuration et administration d'un serveur OpenLdap



# Le protocole LDAP

- Comment s'établit la communication client-serveur
  - bind, unbind, abandon
- Comment s'établit la communication serveur-serveur
  - synchronisation (replication service)
  - liens entre différents annuaires (referral service)
- Transport des données :
  - pas l'ASCII (http, smtp, ...) mais Basic Encoding Rules (BER)
- Les mécanismes de sécurité
  - Méthodes de chiffrement et d'authentification
  - Mécanismes d'accès aux données
- Les opérations de base
  - search, add, delete, etc.

# Plan

- Introduction
- Protocole
- **Modèle d'information**
- Modèle de nommage
- Modèle fonctionnel
- Modèle de sécurité
- Modèle de duplication
- Conception d'un annuaire
- Architecture
- Configuration et administration d'un serveur OpenLdap

# Le modèle d'information

- Le modèle d'information définit le type des données pouvant être stockées dans l'annuaire
- Les éléments composants ce modèle d'information sont :
  - La base de données
  - L'entrée
  - Les attributs
  - Le Schéma
  - L'objectClass
  - Le format LDIF

# La base de données

- Un modèle hiérarchique :
  - racine
  - pays
  - organisation
  - unité d'organisation (ou)
    - « nom commun » (cn)
- Une unité de base : l'objet.
- Un objet est un ensemble indissociable de valeurs.

# L'entrée

- Élément de base de l'annuaire
- Contient les informations sur un objet de l'annuaire
- Ces informations sont représentées sous forme d'un ensemble de paires (attribut, valeur)
- Chaque entrée doit appartenir à une classe particulière
- A chaque attribut est associé un type et une ou plusieurs valeurs
- Les attributs d'une entrée peuvent être obligatoires ou optionnels

# Les attributs (1)

- Un attribut est défini par :
  - son OID.
  - son nom.
  - une courte description de l'attribut.
  - les critères de comparaison utilisés lors d'une recherche.
  - une syntaxe décrivant le type de données.
  - Un attribut peut être multi-valué.
  - Exemple :

```
attributetype ( 2.5.4.5 NAME 'serialNumber'  
DESC 'RFC2256: serial number of the entity'  
EQUALITY caseIgnoreMatch  
SUBSTR caseIgnoreSubstringsMatch  
SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{64} )
```

# Les attributs (2)

- On distingue deux types d'attribut :
  - Les attributs utilisateurs :
    - ils peuvent être modifiés par les utilisateurs.
    - exemple : mail, cn, telephoneNumber.
  - Les attributs opérationnels :
    - ils sont liés au fonctionnement de l'annuaire.
    - ils ne sont pas accessibles aux utilisateurs
    - (exemple : modifytimestamp).
- Un attribut peut hériter d'un autre :
  - l'héritage est défini par la caractéristique SUP.
  - le fils hérite des caractéristiques du père.

# Les attributs (3)

- Les principaux types d'attributs définis par la norme LDAP sont :
  - binary : suite quelconque d'octets (ex: photo).
  - boolean : valeur vrai ou faux.
  - dn : pointeur sur un objet de l'annuaire.
  - Directory String : chaîne de caractères au format UTF-8.
  - integer : valeur entière.
  - telephoneNumber : numéro de téléphone.
- Ils sont identifiés par un OID.



# Les attributs (4)

- Les principales règles de comparaison des attributs sont :
  - `caseIgnoreMatch` : ignorer la casse lors de la comparaison de deux chaînes de caractères.
  - `caseExactMatch` : tenir compte de la casse.
  - `telephoneNumberMatch` : ignorer la casse et supprimer les espaces, virgules, points, etc.
  - `integerMatch` : comparer deux entiers.
  - `booleanMatch` : comparer deux attributs booléens.
  - `distinguishedNameMatch` : comparer des DN.
  - `octetStringMatch` : comparer des binaires octet par octet.

# Les classes d'objets (1)

- Elles décrivent les entrées d'un annuaire.
- Elles sont composées d'attributs.
- Elles définissent un type de «ressource ».
- Elles peuvent être agrégées.
- Elles peuvent hériter d'autres classes.

# Les classes d'objets (2)

- Une classe d'objets est définie par :
  - son OID.
  - son nom.
  - une courte description de la classe.
  - la classe dont elle dérive.
  - son type (ABSTRACT, STRUCTURAL, AUXILIARY).
  - la liste des attributs obligatoires (MUST).
  - la liste des attributs facultatifs (MAY).

```
objectclass ( 2.5.6.6 NAME 'person'  
  DESC 'RFC2256: a person'  
  SUP top STRUCTURAL  
  MUST ( sn $ cn )  
  MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )
```

# Les classes d'objets (3)

- On distingue trois types de classes d'objets :
  - les classes abstraites :
    - elles ne peuvent pas avoir d'instance.
    - seules les classes dérivées peuvent être instanciées.
    - exemple : la classe top dont dérivent toutes les classes d'un annuaire.
  - les classes structurelles :
    - elles peuvent être instanciées.
    - exemple : la classe person dont on trouve des instances dans un annuaire.
  - les classes auxiliaires :
    - elles étendent les classes de type structurel.
    - elles dérivent directement de la classe top.

# Les classes d'objets (4)

The screenshot shows the LDAP Editor interface. On the left, a directory tree is visible with the following structure:

- cyrodil
  - Searches
  - Directory
    - cn=Subschema
    - dc=litislab,dc=eu
      - admin
      - CYRODIL
      - litis-lehavre
      - litis
        - people
          - Julien Baudry

The right pane displays a table of object attributes for the selected entry:

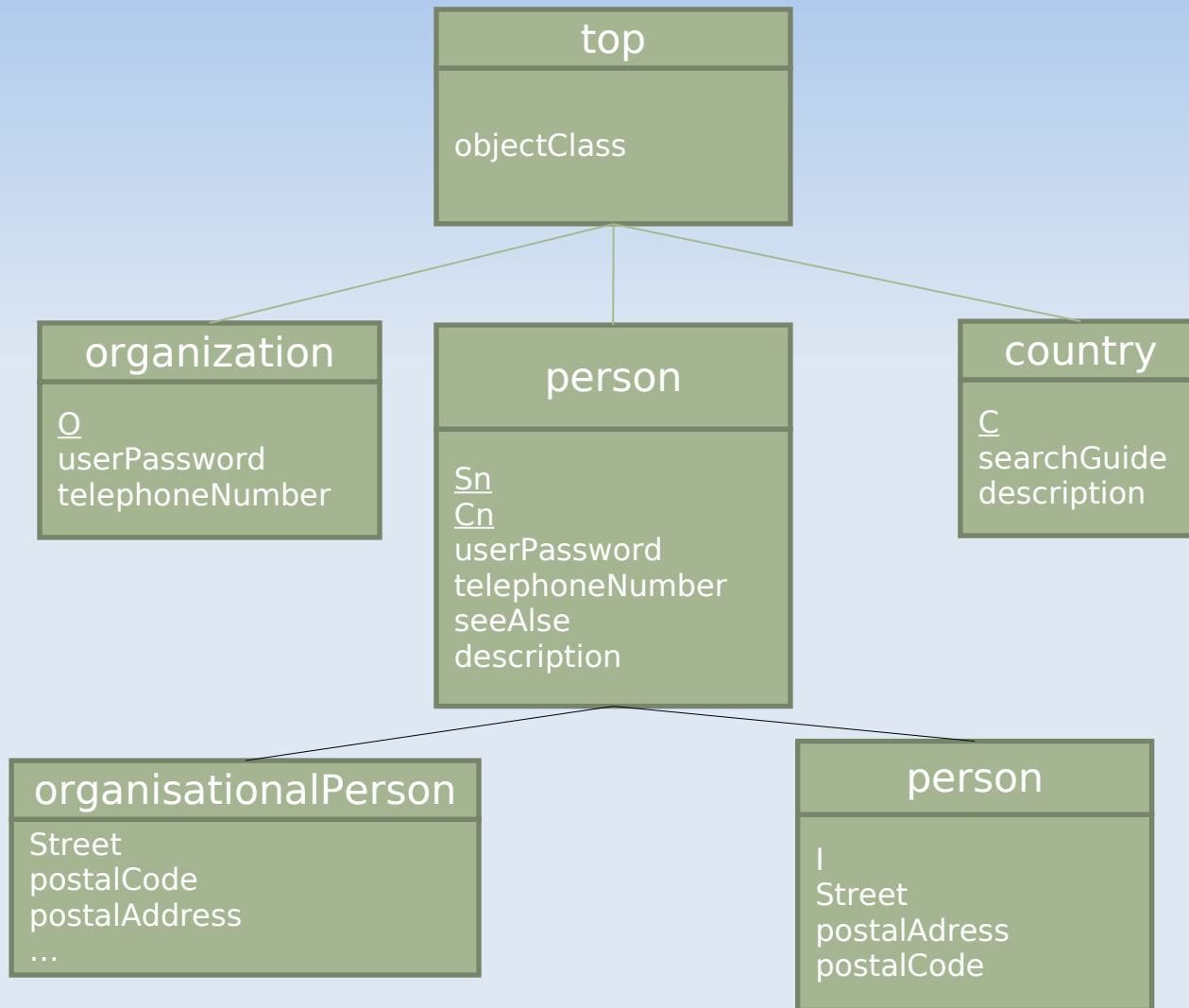
Name	Value	Type
supannOrganisme	LITIS	String
cn	Julien BAUDRY	String
objectClass	inetOrgPerson	String
objectClass	organizationalPerson	String
objectClass	person	String
objectClass	top	String
objectClass	supannPerson	String
objectClass	eduPerson	String
userPassword		String
supannAffectation	RECHERCHE UNIVERSITE LE HAVRE	String
supannAffectation	Université du Havre	String
sn	BAUDRY	String
title	Ingénieur de Recherche	String
uid	jbaudry	String
supannListeRouge	FALSE	String
telephoneNumber	02 32 74 43 73	String
mail	Julien.Baudry@univ-lehavre.fr	String
eduPersonAffiliation	Ingénieur Conception	String
manager	cn=BERTELLE Cyrille	String
givenName	Julien	String

At the bottom of the window, a status bar indicates "Returned: 291 bytes" and "Logged in as cn=admin,dc=litislab,dc=eu".

# L'héritage entre classes d'objets (1)

- L'héritage au sens LDAP est défini ainsi :
  - une classe ne peut dériver que d'une seule classe (pas d'héritage multiple).
  - une classe peut avoir plusieurs filles.
  - toutes les classes dérivent de la classe abstraite top.
- la classe top ne possède qu'un seul attribut :
  - l'attribut obligatoire objectClass.
  - un attribut ne peut pas être surclassé.

# L'héritage entre classes d'objets (2)



# L'agrégation d'objets (1)

- Une entrée de l'annuaire peut être constituée de plusieurs classes d'objets.
  - L'une de ces classes doit être de type structurel.
- Les attributs obligatoires sont la somme des attributs obligatoires de chacune de ces classes.
  - Un attribut commun à plusieurs classes sera « partagé ».



# L'agrégation d'objets (2)

The screenshot shows the LDAP Editor interface. On the left, a directory tree is visible with the following structure:

- cyrodil
  - Searches
  - Directory
    - cn=Subschema
    - dc=litislab,dc=eu
      - admin
      - CYRODIL
      - litis-lehavre
      - litis
        - people
          - Julien Baudry

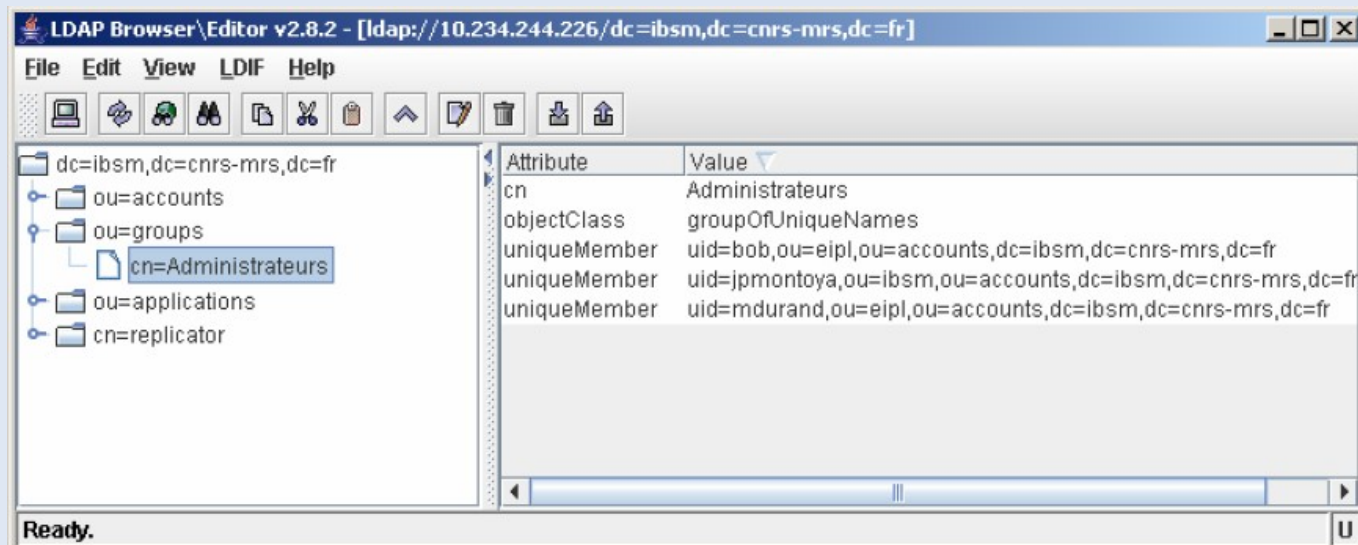
The right pane displays a table of LDAP object attributes for the selected entry:

Name	Value	Type
supannOrganisme	LITIS	String
cn	Julien BAUDRY	String
objectClass	inetOrgPerson	String
objectClass	organizationalPerson	String
objectClass	person	String
objectClass	top	String
objectClass	supannPerson	String
objectClass	eduPerson	String
userPassword		String
supannAffectation	RECHERCHE UNIVERSITE LE HAVRE	String
supannAffectation	Université du Havre	String
sn	BAUDRY	String
title	Ingénieur de Recherche	String
uid	jbaudry	String
supannListeRouge	FALSE	String
telephoneNumber	02 32 74 43 73	String
mail	Julien.Baudry@univ-lehavre.fr	String
eduPersonAffiliation	Ingénieur Conception	String
manager	cn=BERTELLE Cyrille	String
givenName	Julien	String

At the bottom of the window, a status bar indicates "Returned: 291 bytes" and "Logged in as cn=admin,dc=litislab,dc=eu".

# Les relations entre objets

- Elles sont définies par un attribut de type DN.
- Cet attribut contient un pointeur vers un autre objet de l'annuaire.
- Cela permet de se rapprocher du modèle relationnel des bases de données.

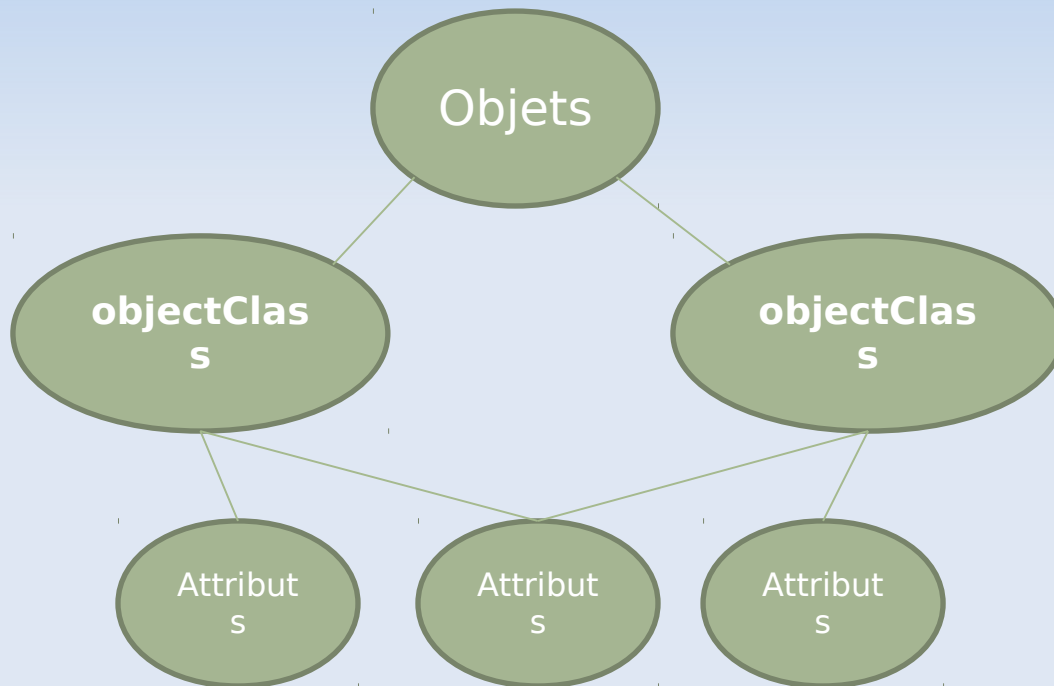


# Le schéma (1)

- Comment savoir quels sont les objectClass disponibles et quels attributs ils contiennent?
- Il est constitué de l'ensemble :
  - des attributs.
  - de leurs syntaxes.
  - des règles de comparaison.
  - des classes d'objets.
- Il est défini dans l'annuaire à l'aide d'attributs et de classes spécifiques.
- Il permet de garantir la validité et l'intégrité des données.

# Le schéma (2)

- Structure d'un schéma



# Le schéma (3)

- Exemple de schéma :

```
attributetype ( 2.5.4.5 NAME 'serialNumber'  
  DESC 'RFC2256: serial number of the entity'  
  EQUALITY caseIgnoreMatch  
  SUBSTR caseIgnoreSubstringsMatch  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{64} )
```

```
attributetype ( 2.5.4.6 NAME ( 'c' 'countryName' )  
  DESC 'RFC2256: ISO-3166 country 2-letter code'  
  SUP name SINGLE-VALUE )
```

```
attributetype ( 2.5.4.9 NAME ( 'street' 'streetAddress' )  
  DESC 'RFC2256: street address of this object'  
  EQUALITY caseIgnoreMatch  
  SUBSTR caseIgnoreSubstringsMatch  
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )
```

```
attributetype ( 2.5.4.11 NAME ( 'ou' 'organizationalUnitName' )  
  DESC 'RFC2256: organizational unit this object belongs to'  
  SUP name )
```

# Le format LDIF (1)

- LDIF signifie LDAP Data Interchange Format.
- Il s'agit du format d'échange pour les annuaires LDAP.
- Il est basé sur un format texte ASCII.
- Il permet d'importer ou d'exporter des données :
  - création,
  - mise à jour,
  - réplication.

# Le format LDIF (2)

- Les entrées de l'annuaire sont décrites sous forme de blocs indépendants.
- Chaque entrée/bloc est séparé par une ligne vide.
- Chaque bloc commence par le DN de l'objet.
- Le RDN doit également se trouver dans la liste des attributs.

# Le format LDIF (3)

dn: **ou=accounts, dc=ibsm,dc=cnrs-mrs,dc=fr**

**ou: accounts**

objectClass: top

objectClass: organizationalUnit

dn: **ou=ibsm,ou=accounts, dc=ibsm,dc=cnrs-mrs,dc=fr**

**ou: ibsm**

objectClass: top

objectClass: organizationalUnit

dn: **uid=jdoe,ou=ibsm,ou=accounts, dc=ibsm,dc=cnrs-mrs,dc=fr**

givenName: John

objectClass: posixAccount

objectClass: top

objectClass: shadowAccount

**userPassword:: e2NyeXB0fWFOQTdvNHBNS2J3dmM=**

**uid: jdoe**

mail: jdoe@ibsm.cnrs-mrs.fr

uidNumber: 1612

cn: John Doe

loginShell: /bin/false

gidNumber: 600

homeDirectory: /home/vmail/jdoe

sn: Doe



# Plan

- Introduction
- Protocole
- Modèle d'information
- **Modèle de nommage**
- Modèle fonctionnel
- Modèle de sécurité
- Modèle de duplication
- Conception d'un annuaire
- Architecture
- Configuration et administration d'un serveur OpenLdap

# Modèle de nommage

- **Le modèle de nommage est la manière dont sont organisées les données dans l'annuaire.**
- LDAP organise les données de manière hiérarchique dans l'annuaire
- Cette arborescence est liée au nommage de chaque élément : un élément marque son appartenance à l'élément supérieur en en reprenant le nom, qu'il complète par le sien.
- Directory Information Tree (DIT)
  - Les entrées gérées par le serveur LDAP sont toutes nommées
  - L'espace de nommage est organisé sous la forme d'un arbre
  - LDAP ne permet pas de limiter les relations de contenance entre classes d'objets : tout est permis.

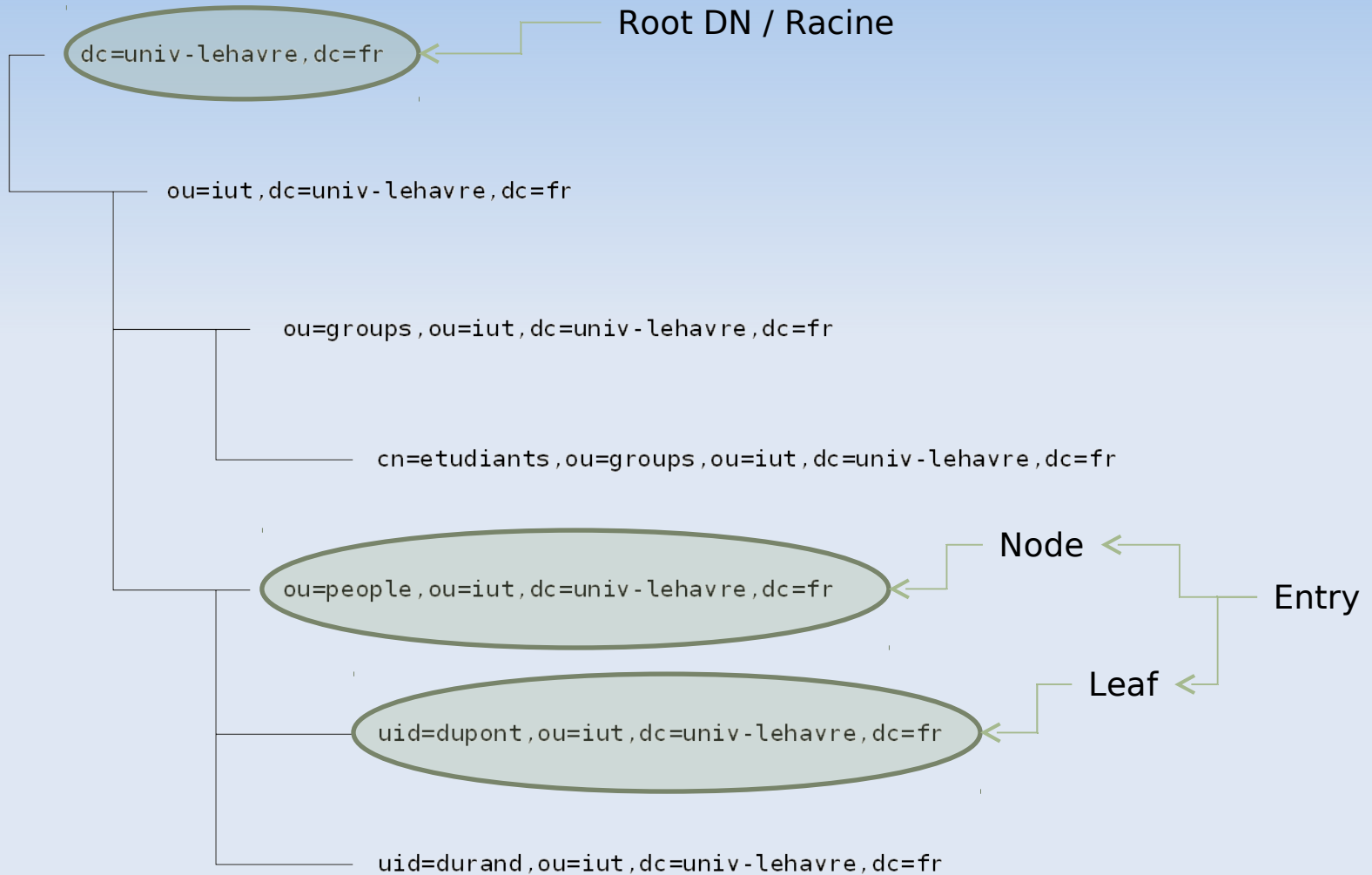
# Quelques termes (1)

- Chaque élément est appelé une **entrée** (an entry).
  - Une entrée peut être un branchement (un **noeud**, a node) ou un élément terminal (une **feuille**, a leaf).
- Chaque élément possède un **DN** (Distinguished Name).
  - Le DN est le nom complet de l'élément qui permet de le positionner dans l'arborescence. Il est unique dans l'annuaire.
    - Exemple : "cn=etudiants,ou=groups,ou=iut,dc=univ-lehavre,dc=fr "

# Quelques termes (2)

- Chaque élément possède également un **RDN** (Relative Distinguished Name).
  - Le RDN est la partie du **DN** de l'élément qui est relative au **DN** supérieur.
  - Le RDN d'un élément ne permet pas de l'identifier de manière absolue dans l'annuaire.
    - Exemple : "cn=etudiants"
- La **racine** est l'élément supérieur de tous les autres, c'est la base de l'arborescence. On l'appelle **root** en anglais, parfois on parle de "**root DN**".
  - Exemple : "dc=univ-lehavre,dc=fr"

# Arborescence



# Les OID : Object Identifier (1)

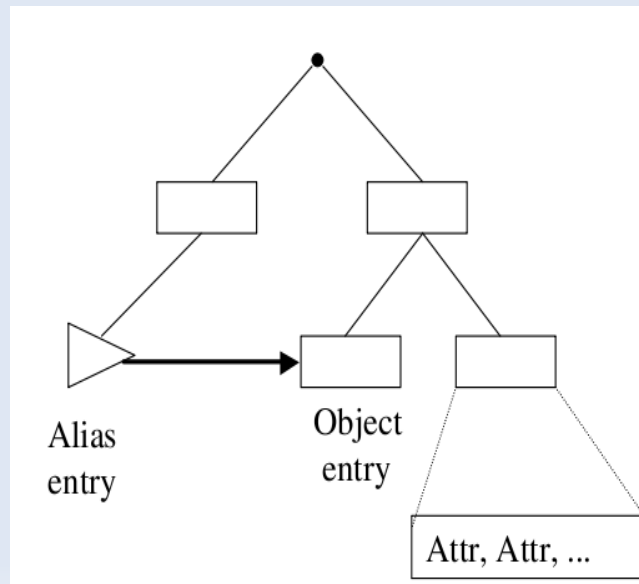
- Un OID est un identifiant unique associé à :
  - chaque classe d'objet.
  - chaque type d'attribut.
- Un OID est composé de plusieurs numéros séparés par un point.
- Chaque numéro représente une branche d'un arbre hiérarchique.
- Tous les attributs du standard commencent par 2.5.4.
- Toutes les classes d'objet commencent par 2.5.6.

# Les OID : Object Identifier (2)

- Les numéros sont affectés par une instance de normalisation : IANA ou ANSI.
- Certaines organisations se voient déléguer l'attribution des numéros pour une sous branche :
  - Standard LDAP (1.3.6.1.4.1.1466.101.120).
  - Université du Michigan (1.3.6.1.4.1.250.1, 2 ou 3).
  - Microsoft pour AD (1.2.840.113556.1).
  - ...etc.
- Un site recense les OID normalisés :
  - <http://www.alvestrand.no/objectid>

# Alias

- Deux types d'objets particuliers :
  - Alias
  - Referrals
- Alias : référence entre entrées au sein d'un même annuaire

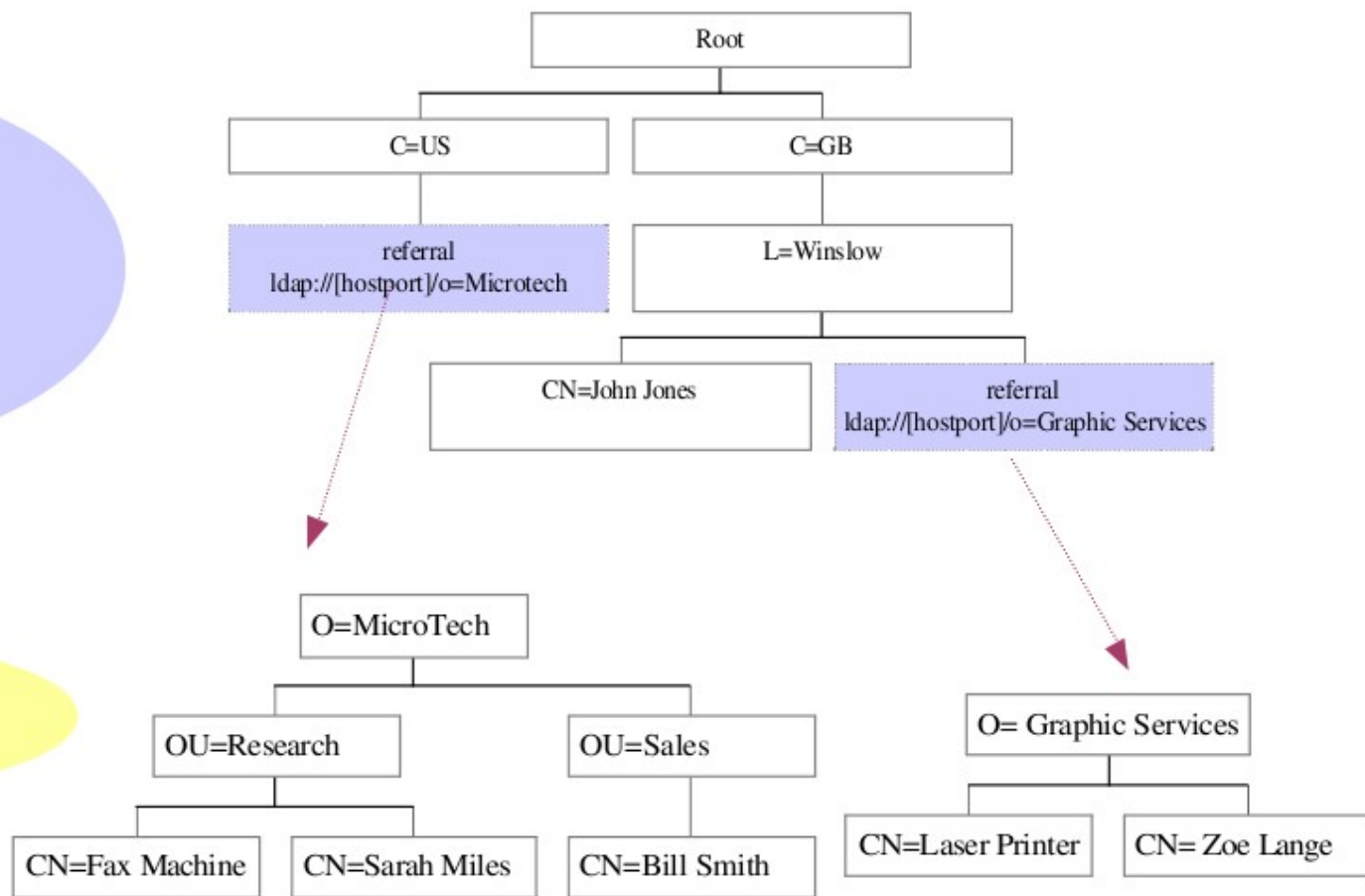
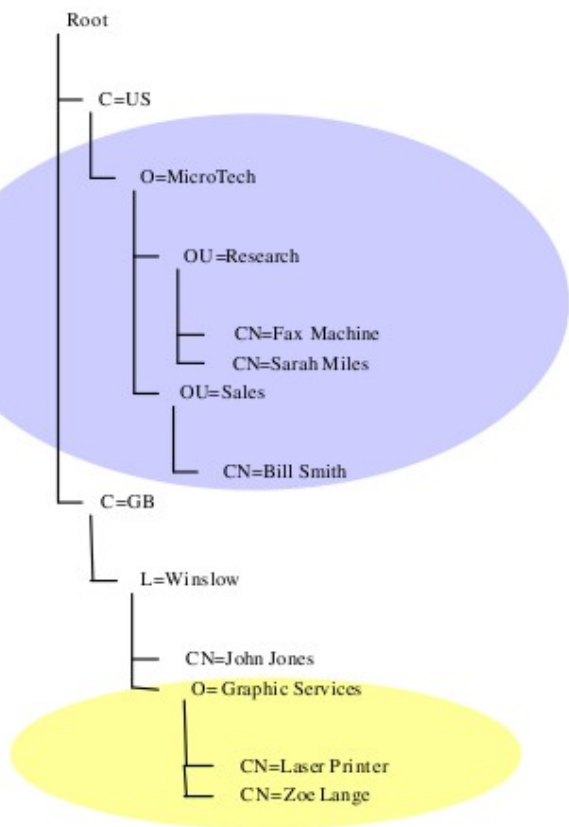




# Referrals

- Referrals :
  - Distribuer la gestion d'un annuaire entre plusieurs serveurs LDAP distincts
  - Chaque serveur gère un sous-ensemble du DIT global
  - Permet la montée en charge en gardant de bonnes performances
- Gestion de la distribution
  - La distribution est gérée au niveau du client LDAP, il est responsable de toutes les connexions
  - Permet dans l'Internet de préserver l'autonomie des serveurs car :
    - La bande passante entre un client et serveur et la même qu'entre serveur et serveur
    - Les clients sont suffisamment puissants
    - Limite : tout le travail incombe à l'utilisateur

# Referrals



# Plan

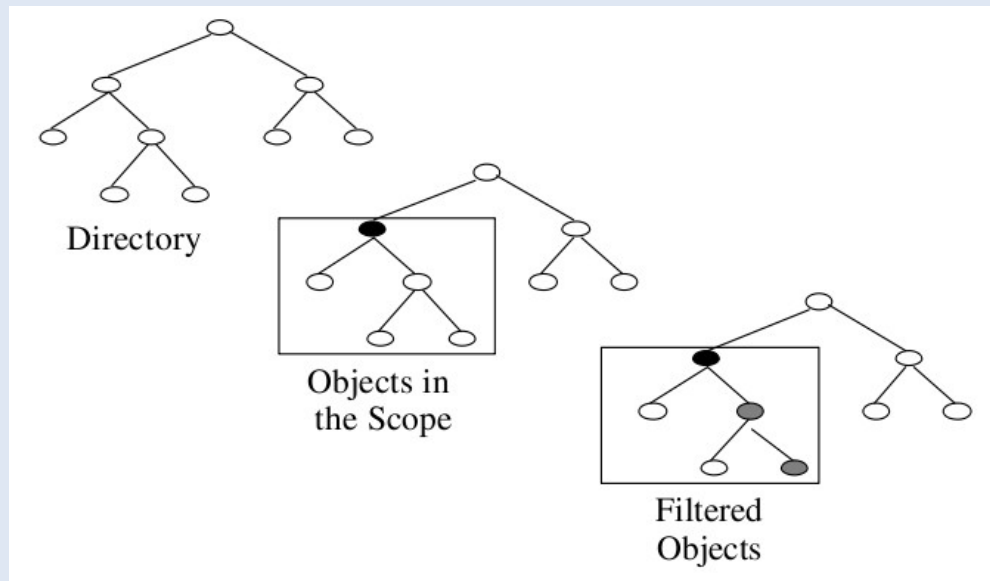
- Introduction
- Protocole
- Modèle d'information
- Modèle de nommage
- **Modèle fonctionnel**
- Modèle de sécurité
- Modèle de duplication
- Conception d'un annuaire
- Architecture
- Configuration et administration d'un serveur OpenLdap

# Le Modèle fonctionnel

- Décrit le moyen d'accéder aux données ainsi que les opérations qu'on peut leur appliquer
- Le modèle définit :
  - les opérations d'interrogation
  - les opérations de comparaison
  - les opérations de mise à jour
  - les opérations d'authentification et de contrôle

# Interrogation de l'annuaire

- Interrogation
  - LDAP ne fournit pas d'opération de lecture d'entrée
  - Pour connaître le contenu d'une entrée, il faut écrire une requête
- Scope & Filter

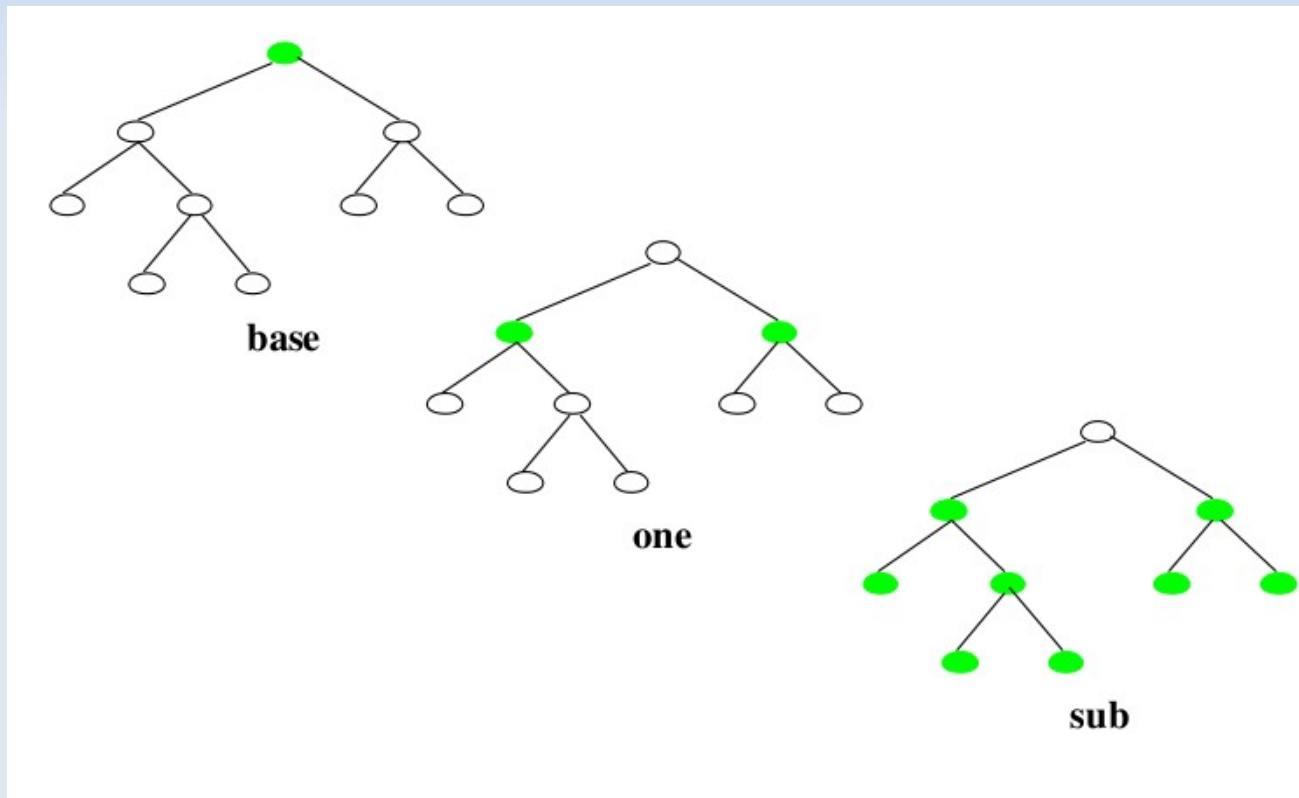


# Interrogation de l'annuaire : la portée

- **scope=base**
  - Permet de faire une recherche dans la base de l'annuaire, et retourne toutes les entrées trouvées.
- **scope=sub**
  - Permet de faire une recherche à partir du noeud spécifié par la base, dans le noeud courant, mais aussi dans tous les noeuds sous jacents : recherche récursive dans toutes les branches du sous-arbre prenant racine à la base considérée
- **scope=one**
  - Permet de faire une recherche à partir du noeud spécifié par la base, uniquement sur le niveau courant. Cela évite de descendre visiter les OU s'il n'y en a pas besoin.

# Interrogation de l'annuaire : la portée

- L'argument -s définit la portée de recherche dans la base.
- Le scope définit la profondeur de la recherche dans le DIT.



# Interrogation de l'annuaire : le filtre

- Filter :
  - Permet de tester l'existence ou les valeurs d'attributs
  - Exemple :
    - `objectclass = person`
    - `telephonenumber = 01*`
  - La clause filter d'une requête LDAP est composé d'un ensemble de conjonctions et/ou de disjonctions de filtres simples



# Interrogation de l'annuaire : Idapsearch

- Forme de requete la plus simple
  - `Idapsearch -x -LLL [filtre] [attributs a retourner]`
- Les arguments de Idapsearch:
  - `-x` : pas d'authentification sasl
  - `-LLL` : sorties au format ldif
  - `-h` : désigner le serveur ldap
  - `-b` : le DN de la base
  - `-D` : connexion authentifiée à la base
  - `-W -w` : en donnant le mot de passe
  - `-s` : la portée de recherche dans la base

# Interrogation de l'annuaire : Idapsearch

- Forme de requete la plus simple
  - `Idapsearch -x -LLL [filtre] [attributs a retourner]`
  - `Idapsearch -x objectclass=posixaccount sn`
- Interrogation avec une connexion ( bind) en « admin »
  - `Idapsearch -x -LLL -D uid=admin,dc=iut,dc=univ-lehavre,dc=fr -Wuid=toto`
- Interrogation sur une branche particulière
  - `Idapsearch -LLL -x -b ou=groups,dc=iut,dc=univ-lehavre,dc=fr objectclass=* cn`

# Interrogation de l'annuaire : opérateurs "et" "ou" (1)

- requêtes complexes « et » « ou » : notation préfixée (et(requeteA)(requeteB))
  - Approximation
    - (sn~lbs) ##orthographe voisine de lbs
  - Égalité stricte
    - (sn=toto) ##recherche exactement toto
  - Comparaison
    - (sn>toto) , <= , >= , < ##noms situés alphabétiquement après ou avant toto
  - Présence
    - (sn=\*) ##retourne les entrées ayant un attribut sn présent

# Interrogation de l'annuaire : opérateurs "et" "ou" (2)

- Requêtes complexes « et » « ou » : notation préfixée (et(requeteA)(requeteB))
  - Sous-chaîne
    - (sn=to\*), (sn=\*to), (sn=t\*t\*) ##expressions régulières sur les chaînes
  - ET
    - (&(sn=toto) (u2site=luminy)) ##nom est toto et du site luminy
  - OU
    - '!(sn~=lbs)(sn=gora))' sn ## nom proche de lbs ou gora
  - Négation
    - (!(tel=\*)) toutes les entrées sans attribut téléphone

# Interrogation de l'annuaire : ET OU

- Les personnels de l'IUT
  - `ldapsearch -h annuaire.iut-lehavre.fr -L -x -b "ou=people,dc=univ-lehavre,dc=fr" '(&(supannAffectation=IUT)(objectClass=supannPersonne))' cn`
- Les personnels du l'IUT des sites Caucriauville OU Frissard
  - `ldapsearch -L -x -b "ou=people,dc=iut,dc=univ-lehavre,dc=fr" '(&(supannAffectation=iut)(site=Caucriauville)(Site=Frissard))' cn`
- attention aux parenthèses et à la syntaxe

# Interrogation de l'annuaire : non !

- requêtes complexes « non » (!(A))
  - Tous les comptes de la base sauf ceux du site Frissard
- `ldapsearch -L -x -b "ou=people,dc=com,dc=univ-lehavre, dc=fr" '(&(objectclass=posixaccount)(!(u2Site=Frissard)))' uid cn u2Site`

# Plan

- Introduction
- Protocole
- Modèle d'information
- Modèle de nommage
- Modèle fonctionnel
- **Modèle de sécurité**
- Modèle de duplication
- Conception d'un annuaire
- Architecture
- Configuration et administration d'un serveur OpenLdap

# Modèle de sécurité (1)

- Elle consiste à :
  - protéger l'accès aux données :
  - chiffrement des transferts entre le serveur et les clients.
  - politique de listes de contrôle d'accès.
  - filtrage au niveau TCP.
  - préserver l'intégrité des données :
  - mise en œuvre de dispositifs de réplication.
  - prévenir les dénis de services (DOS).



# Modèle de sécurité (2)

- Décrit le moyen de protéger les données de l'annuaire des accès non autorisés
- Plusieurs niveaux :
  - authentification lors de l'accès à un service
    - anonymous permet de consulter les données accessibles en lecture pour tous
    - administrateur (tous les droits)
    - mot de passe en clair (DN + password transigent en clair sur le réseau)
    - Mot de passe + SSL ou TLS (la session est chiffrée)
    - Échange de certificats SSL (clés publiques/privées)

# Modèle de sécurité (3)

- Plusieurs niveaux :
  - authentification lors de l'accès à un service
  - Simple Authentication and Security Layer (SASL) :  
mécanisme externe d'authentification (Kerberos, S/Key, GSSAPI)
  - contrôle d'accès
  - définit les droits des différents utilisateurs sur les données
  - chiffrement des transactions entre clients et serveurs ou entre serveurs

# Les droits d'accès (1)

- Les droits d'accès :
  - permettent de gérer les autorisations sur la totalité des entrées de l'annuaire.
  - s'appliquent sur les objets et sur leurs attributs.
  - consistent à décrire les droits de certains objets de l'annuaire sur d'autres entrées.
- Cette description s'effectue à l'aide de règles (ACL).
  - Chaque ACL comprend plusieurs règles (ACI).
  - La syntaxe d'une ACI n'est pas normalisée.
  - Les ACL du serveur « slapd »
    - Access to <un attribut>
    - By <uid de la base> <type d'autorisation>
    - By <uid de la base> <type d'autorisation>

# Les droits d'accès (2)

- Les listes de contrôle d'accès répondent aux questions suivantes :
  - Qui ?
- anonyme (anonymous), utilisateur (self), groupé d'utilisateurs (users), tout le monde (\*).
  - A partir d'où ?
- nom de machine ou adresse IP source.
- Quels droits ?
  - authentification (auth), lecture (read), écriture (write), suppression (write), ajout (write), recherche (search), comparaison (compare).
- Sur quoi ?
  - attribut, objet, totalité de l'annuaire (\*).

# Plan

- Introduction
- Protocole
- Modèle d'information
- Modèle de nommage
- Modèle fonctionnel
- Modèle de sécurité
- Modèle de duplication
- **Conception d'un annuaire**
- Architecture
- Configuration et administration d'un serveur OpenLdap

# La conception (1)

- Elle est l'étape la plus importante du processus de mise en oeuvre de l'annuaire.
- Elle se décompose en plusieurs phases que l'on peut résumer ainsi :
  - Quelles informations mettre dans l'annuaire ?
    - (définition des attributs et des classes d'objets).
  - Quelle est la provenance des données et quels en sont les propriétaires ?
  - Comment les organiser dans un modèle commun à toutes les applications ?
    - (schéma de l'annuaire, organisation hiérarchique).

# La conception (2)

- Qui va gérer ces données et de quelle manière ?
- Comment les faire évoluer dans le temps ?
  - (définition des attributs et des classes supplémentaires).
- Quelle politique d'accès appliquer ?
- Quelles applications clientes les utiliseront ?

# La conception (3)

- Il est indispensable de :
  - bien prendre en compte tous les acteurs.
  - suivre scrupuleusement la procédure de déploiement pour détecter les incohérences et les problèmes le plus en amont possible.
- Quelques règles s'imposent :
  - démarrer petit,
  - grossir petit à petit,
  - valider sur un annuaire qui n'est pas en production,
  - rester simple dans la conception.



# L'étape de cadrage

- Elle consiste à identifier les besoins qui motivent la mise en place d'un annuaire.
- Il est conseillé de :
  - limiter le champ de l'annuaire à une ou deux applications.
  - choisir une application représentative des besoins les plus larges.
  - justifier la pertinence de l'intégration de cette application à l'annuaire.
- Quelques exemples d'applications :
  - authentification auprès d'un service de messagerie.
  - annuaire de type « Pages Blanches ».

# L'élaboration du contenu (1)

- L'objectif de cette étape est de définir :
  - les attributs,
  - les classes d'objets,
  - la hiérarchie entre ces classes.
- Quelques exemples de données :
  - personnes : nom, prénom, téléphone, fonction, etc.
  - organisations : nom, adresse, contacts, etc.
  - ressources : ordinateur, téléphone portable, etc.
  - paramètres de configuration d'applications.
  - etc.

# L'élaboration du contenu (2)

- Les informations stockées dans l'annuaire doivent être pérennes.
  - ex : identifiants, mots de passe, adresses.
- Elles doivent intéresser plusieurs « applications clientes ».
- Il ne faut pas confondre un annuaire avec :
  - une base de données relationnelle.
  - un système de fichiers.
  - un serveur Web ou un serveur FTP.

# La description des données (1)

- La description des données découle des applications qui utiliseront l'annuaire.
- Il est recommandé d'anticiper les évolutions futures de l'annuaire.
- Il faut recenser les données :
  - actuelles.
  - futures en fonction des applications de l'entreprise.
  - celles qu'il serait intéressant d'ajouter à l'annuaire selon vous.

# La description des données (2)

- Quelles sont les sources des données actuelles ?
  - serveur de messagerie,
  - applications de ressources humaines,
  - ...
- Quels seront les outils utilisés pour importer les données dans l'annuaire ?
  - applications existantes,
  - développement d'outils spécifiques en interne,
  - ...
- Quel sera le référentiel principal ?

# L'accès aux données

- On définit la politique d'accès aux données pour les utilisateurs et les applications.
- Il est possible de déléguer la gestion des données à un utilisateur ou à un groupe.
  - exemple : mise à jour du numéro de téléphone.
- Exemples :
  - lecture seule pour tout le monde,
  - accès complet pour un groupe d'administrateurs,
  - chaque utilisateur a le droit de modifier certains attributs,
  - accès en écriture sur un sous-ensemble de données pour certains groupes de personnes,
  - ...

# Conception du schéma de l'annuaire

- La conception du schéma permet de définir :
  - les attributs obligatoires,
  - les attributs facultatifs/autorisés,
  - les classes d'objets,
  - les DN,
  - la structure de l'annuaire.

# La définition des attributs (1)

- Elle s'effectue parallèlement à la définition des classes.
- On liste les principaux attributs communs à l'ensemble des classes d'objets.
- Chaque donnée est associée à un attribut.
- Suivant la nature de la donnée, il est conseillé de normaliser les valeurs.
  - exemple : "04.91.16.43.33" ou "04 91 16 43 33".
- Ces conventions devront être respectées par les applications clientes.



# La définition des attributs (2)

- On utilise autant que possible les attributs définis par la norme LDAP.
- On doit respecter la sémantique des attributs standards.
  - ex : attribut pays "c" codé sur 2 lettres (ISO 3166).
- On définit si l'attribut sera multi-valué.
- On établit la fréquence de lecture de l'attribut pour éventuellement l'indexer.

# La définition des classes d'objets

- Leur définition dépend :
  - du type de ressource (personne, bureau, machine, etc.).
  - des attributs obligatoires,
  - des attributs autorisés.
- On utilise autant que possible les classes définies par la norme LDAP.
- Les classes d'objets doivent dériver autant que possible des classes du standard LDAP.

# Personnalisation du schéma

- Le schéma d'un annuaire doit souvent être étendu pour :
  - stocker des données non prévues dans le schéma de base.
    - ex : adresse d'une photo, code métier.
- gérer des types de ressources non standards.
  - ex : ordinateur, véhicule.
- ATTENTION !! Ne jamais modifier le schéma standard !
- 5 étapes pour étendre un schéma :
  - Obtenir un Object Identifier
  - Créer un schema local (include)
  - Définir des attributs
  - Définir des ObjectClass

# Objectif

- Nous souhaitons stocker nos comptes de messagerie dans un annuaire LDAP.
- Nous voulons définir pour ces comptes des attributs propres à notre organisation.
- Ce référentiel sera utilisé ultérieurement pour authentifier les utilisateurs auprès d'autres applications :
  - applications Intranet.
  - serveurs FTP.
  - synchronisation avec les domaines Windows.
  - etc.

# Description des données (1)

- Déterminons les attributs qui caractérisent le compte de messagerie d'un utilisateur :
  - l'identifiant,
  - le mot de passe,
  - son adresse de messagerie,
  - son ou ses alias de messagerie,
  - son répertoire de base,
  - le chemin d'accès à sa boîte aux lettres,
  - le protocole de téléchargement des messages stockés sur le serveur,
  - le quota de la BAL exprimé en volume,
  - le quota de la BAL exprimé en nombre de messages,
  - la date d'expiration du compte de l'utilisateur,
  - un champ indiquant si le compte est actif.

# Description des données (2)

- Définissons ensuite les attributs qui décrivent une personne de notre organisation :
  - son nom de famille,
  - son prénom,
  - sa photo,
  - son unité ou équipe,
  - son chef d'équipe,
  - son statut,
  - son bureau,
  - son adresse postale complète,
  - son numéro de téléphone,
  - son numéro de fax,
  - son titre (différent de la fonction),
  - la date d'expiration du compte de l'utilisateur,
  - son certificat électronique,
  - son ou ses ordinateurs.

# Conception du schéma (1)

- Faisons le point sur les attributs fournis par les classes d'objets standards :
  - classe d'objets inetOrgPerson :

Classe	Attributs obligatoires	Attributs optionnels
inetOrgPerson	objectClass <b>sn</b> cn	audio businessCategory carLicense departmentNumber displayName employeeNumber <b>employeeType</b> <b>givenName</b> homePhone homePostalAddress initials <b>jpegPhoto</b> labeledURI <b>mail</b> <b>manager</b> mobile o pager photo <b>roomNumber</b> secretary uid userCertificate x500uniqueIdentifier preferredLanguage <b>userSMIMECertificate</b> userPKCS12

# Conception du schéma (2)

- classe d'objets organizationalPerson :

Classe	Attributs obligatoires	Attributs optionnels
<b>organizationalPerson</b>	objectClass cn sn	description destinationIndicator facsimileTelephoneNumber internationalISDNNumber <b>l</b> ou physicalDeliveryOfficeName <b>postalAddress</b> <b>postalCode</b> <b>postOfficeBox</b> preferredDeliveryMethod registeredAddress seeAlso st street telephoneNumber teletexTerminalIdentifier <b>telexNumber</b> <b>title</b> <b>userPassword</b> x121Address



# Conception du schéma (3)

- classe d'objets posixAccount :

Classe	Attributs obligatoires	Attributs optionnels
posixAccount	objectClass <b>cn</b> <b>uid</b> <b>uidNumber</b> <b>gidNumber</b> <b>homeDirectory</b>	description <b>gecos</b> <b>loginShell</b> <b>userPassword</b>

Classe	Attributs obligatoires	Attributs optionnels
shadowAccount	objectClass <b>uid</b>	description <b>userPassword</b> <b>shadowLastChange</b> shadowMin <b>shadowMax</b> <b>shadowWarning</b> shadowInactive <b>shadowExpire</b> shadowFlag

# Conception du schéma (4)

- Définissons notre classe iutLeHavrePerson :
  - elle hérite de la classe inetOrgPerson.
  - elle est composée de nouveaux attributs qui ont été ajoutés au référentiel commun.



# Conception du schéma (6)

- Définition des attributs :

```
# ibsm.schema, v 0.2 2006-11-02 by TD0
# OID prefix      : 1.3.6.1.4.1.29673
# Attributes      : 1.3.6.1.4.1.29673.1.1
# Objectclasses   : 1.3.6.1.4.1.29673.1.2

attributeType (1.3.6.1.4.1.29673.1.1.1 NAME 'ibsmMailboxPath'
  DESC 'The absolute path to the mailbox'
  EQUALITY caseExactIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE)

attributeType (1.3.6.1.4.1.29673.1.1.2 NAME 'ibsmMailQuotaSize'
  DESC 'The MB size quota related to a mailbox'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE)

attributeType (1.3.6.1.4.1.29673.1.1.4 NAME 'ibsmMailQuotaCount'
  DESC 'The maximum number of allowed messages in the mailbox'
  EQUALITY caseExactIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE)

attributeType (1.3.6.1.4.1.29673.1.1.5 NAME 'ibsmIsAccountActive'
  DESC 'A boolean telling wether the user account is still active)'
  EQUALITY booleanMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE)

attributeType (1.3.6.1.4.1.29673.1.1.6 NAME 'ibsmMailProtocol'
  DESC 'The protocol used to retrieve e-mails from the server)'
  EQUALITY caseIgnoreMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 SINGLE-VALUE)

attributeType (1.3.6.1.4.1.29673.1.1.9 NAME 'ibsmMailAlias'
  DESC 'Identifies the alias(es) of a person'
  EQUALITY caseExactIA5Match
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256})
```

# Conception du schéma (7)

- Définition de la classe d'objets :

```
# ibsm.schema, v 0.2 2006-11-02 by TDO
# OID prefix   : 1.3.6.1.4.1.29673
# Attributes   : 1.3.6.1.4.1.29673.1.1
# Objectclasses : 1.3.6.1.4.1.29673.1.2

objectClass (1.3.6.1.4.1.29673.1.2.1 NAME 'ibsmPerson'
            SUP inetOrgPerson STRUCTURAL
            MUST (ibsmMailBoxPath $ ibsmMailProtocol)
            MAY (ibsmMailQuotaSize $ ibsmMailQuotaCount $ ibsmMailProtocol $
                ibsmIsAccountActive $ ibsmMailAlias))
```

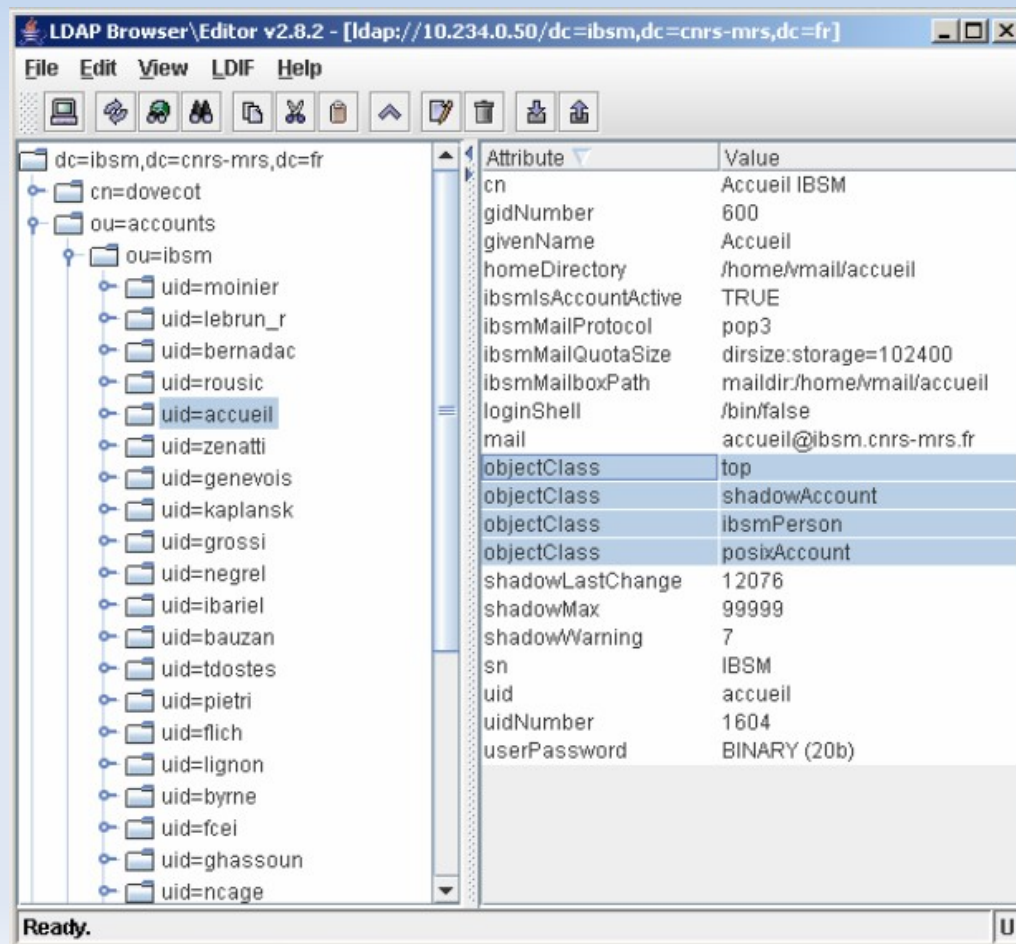
OpenLdap :

- édition du fichier /etc/ldap/slapd.conf :

```
# Schema and objectClass definitions
include      /etc/ldap/schema/core.schema
include      /etc/ldap/schema/cosine.schema
include      /etc/ldap/schema/nis.schema
include      /etc/ldap/schema/inetorgperson.schema
include      /etc/ldap/schema/ibsm.schema
```

# Exemple d'objet

- Définition d'un compte de messagerie :



The screenshot shows the LDAP Browser/Editor v2.8.2 interface. The left pane displays a tree view of the LDAP directory structure, with the following path selected: dc=ibsm,dc=cnrs-mrs,dc=fr > cn=dovecot > ou=accounts > ou=ibsm > uid=accueil. The right pane displays the attributes and values for the selected object:

Attribute	Value
cn	Accueil IBSM
gidNumber	600
givenName	Accueil
homeDirectory	/home/vmail/accueil
ibsmIsAccountActive	TRUE
ibsmMailProtocol	pop3
ibsmMailQuotaSize	dirsize:storage=102400
ibsmMailboxPath	maildir:/home/vmail/accueil
loginShell	/bin/false
mail	accueil@ibsm.cnrs-mrs.fr
objectClass	top
objectClass	shadowAccount
objectClass	ibsmPerson
objectClass	posixAccount
shadowLastChange	12076
shadowMax	99999
shadowWarning	7
sn	IBSM
uid	accueil
uidNumber	1604
userPassword	BINARY (20b)

# Peuplement de l'annuaire OpenLdap (1)

- Création de l'objet racine et des premières entrées.
- Utilisation du format d'échange pour les annuaires LDAP (LDIF).
- Deux méthodes d'importation possibles :
  - en soumettant le fichier au contrôle du serveur d'annuaire (commande `ldapadd`).
  - en exécutant les outils de gestions fournis en natif avec OpenLdap (commande `slapadd`).

# Peuplement de l'annuaire OpenLdap (2)

- Utilisation de la commande ldapadd :

```
ldapadd -W -D <binddn> -x -H ldap://<serveur> -f <fichier.ldif>
```

- -x : authentification simple (au lieu de SASL).
- -W : demande la saisie interactive du mot de passe administrateur de l'annuaire.
- -D : précise le DN de l'utilisateur pour se connecter à l'annuaire.
- -f : le fichier d'échange à importer.

# Peuplement de l'annuaire OpenLdap (3)

- Utilisation de la commande slapadd :

```
slapadd -b dc=ibsm,dc=cnrs-mrs,dc=fr -l base_ldif
```

- Contexte du peuplement :
  - La commande slapadd ne nécessite pas que le serveur d'annuaire soit actif.
  - L'importation se fait sans contrôle d'intégrité au niveau de l'annuaire :
- pas de vérification du schéma.
- pas de contrôle de l'arborescence avant d'ajouter une entrée (DN).
  - pas de mise à jour des attributs opérationnels.



# Annexe

Réplication LDAP  
avec slurpd

# Réplication du serveur OpenLdap

- La mise en oeuvre de la réplication permet:
  - d'optimiser la gestion de la charge.
  - d'améliorer la disponibilité de l'annuaire.
  - de sécuriser les données.
- Il existe deux mécanismes de réplication entre annuaires LDAP :
  - le processus basé sur la génération d'un fichier LDIF et géré par un démon dédié (slurpd).
  - un mécanisme dénommé LDAP Sync Replication disponible depuis la version 2.2 d'OpenLdap.

# Réplication avec slurpd (1)

- Les configurations possibles sont de type :
  - maître/esclave(s).
  - multi-maîtres.
- Fonctionnement du mécanisme :
  - génération d'un fichier LDIF par le serveur LDAP.
  - le démon slurpd analyse le contenu de ce fichier.
  - il applique les modifications sur le serveur esclave.
- Une configuration de réplication s'applique à une base de données LDAP.

# Réplication avec slurpd (2)

- Mise en oeuvre :
  - configuration préliminaire du serveur esclave à l'identique de l'annuaire maître :
- schéma,
- racine et arborescence,
- règles de contrôle d'accès (ACL),
- données.

# Réplication avec slurpd (3)

- Mise en oeuvre (suite) :
  - configuration du serveur maître :
- déclaration du fichier journal des répliquions :

```
# Where to store the replica logs for database #1  
repllogfile /var/lib/ldap/repllog
```

- déclaration du serveur esclave :

```
replica uri=ldap://galileo.ibsm.glm:389  
binddn="cn=replicator,dc=ibsm,dc=cnrs-mrs,dc=fr"  
bindmethod=simple  
credentials=mon_mot_de_passe
```

- création de l'utilisateur dédié au processus de répliquion.

# Réplication avec slurpd (5)

- Mise en oeuvre (suite) :
  - synchronisation du contenu de l'annuaire esclave avec celui de l'annuaire maître.
- exemple : utilisation du format d'échange LDIF.
  - activation de la réplication :
- redémarrage du serveur maître.
- redémarrage du ou des serveurs esclaves.
- exécution, sur le serveur maître, de la commande **slurpd** pour activer le processus de réplication.

# Réplication avec slurpd (6)

- Mise en oeuvre (suite) :
  - lancement automatique de la commande slurpd à chaque démarrage du système :
- édition du fichier /etc/default/slapd :

```
# Configure if the slurpd daemon should be started. Possible values:  
# - yes: Always start slurpd  
# - no: Never start slurpd  
# - auto: Start slurpd if a replica option is found in slapd.conf  
(default)  
SLURPD_START=auto
```
- Campagne de tests :
  - modification d'une valeur sur l'annuaire maître.
  - modification d'une valeur sur l'annuaire esclave.

# Réplication avec slurpd (7)

- Illustration : modification d'une valeur sur l'annuaire esclave.
  - le client LDAP soumet une modification à l'annuaire esclave.
  - ce dernier retourne les informations nécessaires (pointeur) pour s'adresser directement au maître.
  - le client soumet la modification au maître.
  - celui-ci applique la modification et l'inscrit dans le fichier journal de réplication.
  - le démon slurpd détecte la mise à jour et transmet un ordre de mise à jour à l'esclave.
  - le serveur esclave applique le changement et notifie slurpd du bon déroulement de l'opération.