# LDAP101:What is LDAP? LDAP Basics.

# Copyright



If you find this document useful or further its distribution, we would appreciate you letting us know.

# What Is LDAP

- Lightweight Directory Access Protocol
  - A descendent of the X.500 OSI Directory Protocol
    - A cross-platform network protocol for communicating with a directory service provider.
    - A flexible data-model.
      - Hierarchical
      - Object Oriented

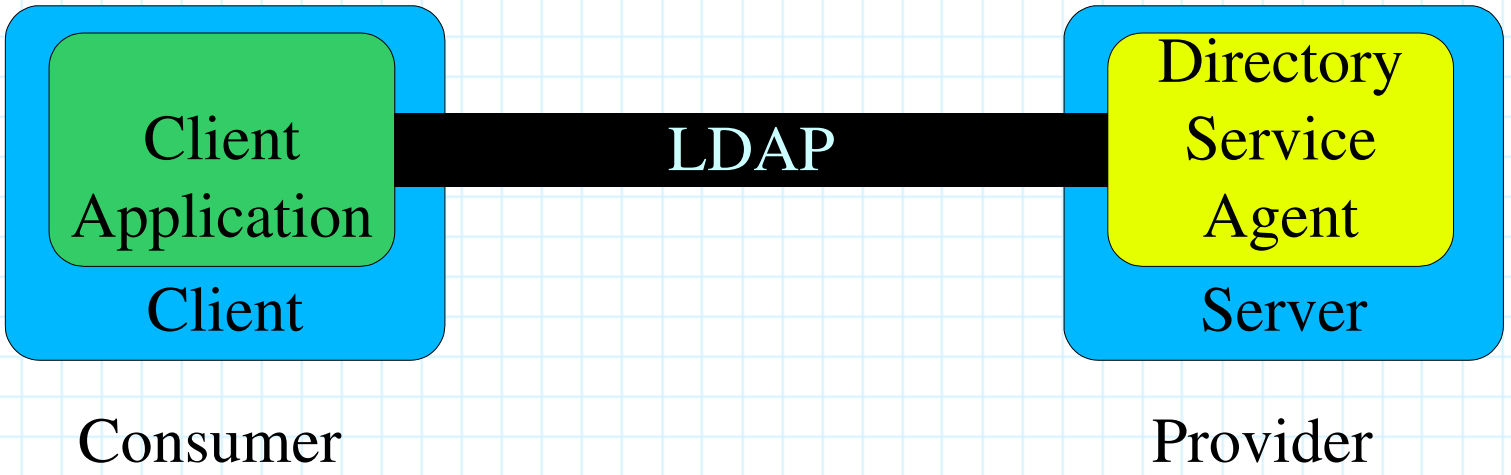
# DSA

- LDAP service is provided by a Directory Service Agent, commonly referred to as a DSA
  - M$-Active Directory
  - OpenLDAP
    - Symas
    - Open Directory v2 (introduced in OS X 10.3)
  - M$-Active Directory
  - IBM SecureWay
  - Lotus Notes
  - Novell E-Directory
  - Oracle Collaberation Suite
  - Netscape Directory Server
  - Sun One Dir
  - Samba 4 (?)

# DSA

- Most LDAP DSAs provide:
  - Replication
  - Granular and context aware access control
    - Who can access the directory
    - When they can access the directory
    - From where they can access the directory
    - Minimal security levels for accessing the directory
  - Partitioning & Referrals.
- May provide:
  - Virtual directory backends.
    - Linking external data sources into your directory.

# Clients & Servers



# What LDAP Is Not

- A replacement for 'traditional' relational database systems.
  - LDAP assumes a very high read to write ratio.
  - LDAP does not natively provide transactional capabilities.
    - A discrete operation is atomic, but operations cannot be grouped into explicit transactions.
- A stand-alone service.
  - LDAP is used as a backend for other network services such as mail and authentication.

# v2 & v3

- There are two versions of LDAP 'in the field'
  - Version 2
  - Version 3
    - Version 3 adds:
      - More powerful schema specification.
      - Schema discovery by clients.
      - TLS support.
      - SASL support.
      - Server-side referrals.
      - Object renaming.
  - Many older applications only support LDAPv2, but LDAPv3 is always preferable.

# Subjects

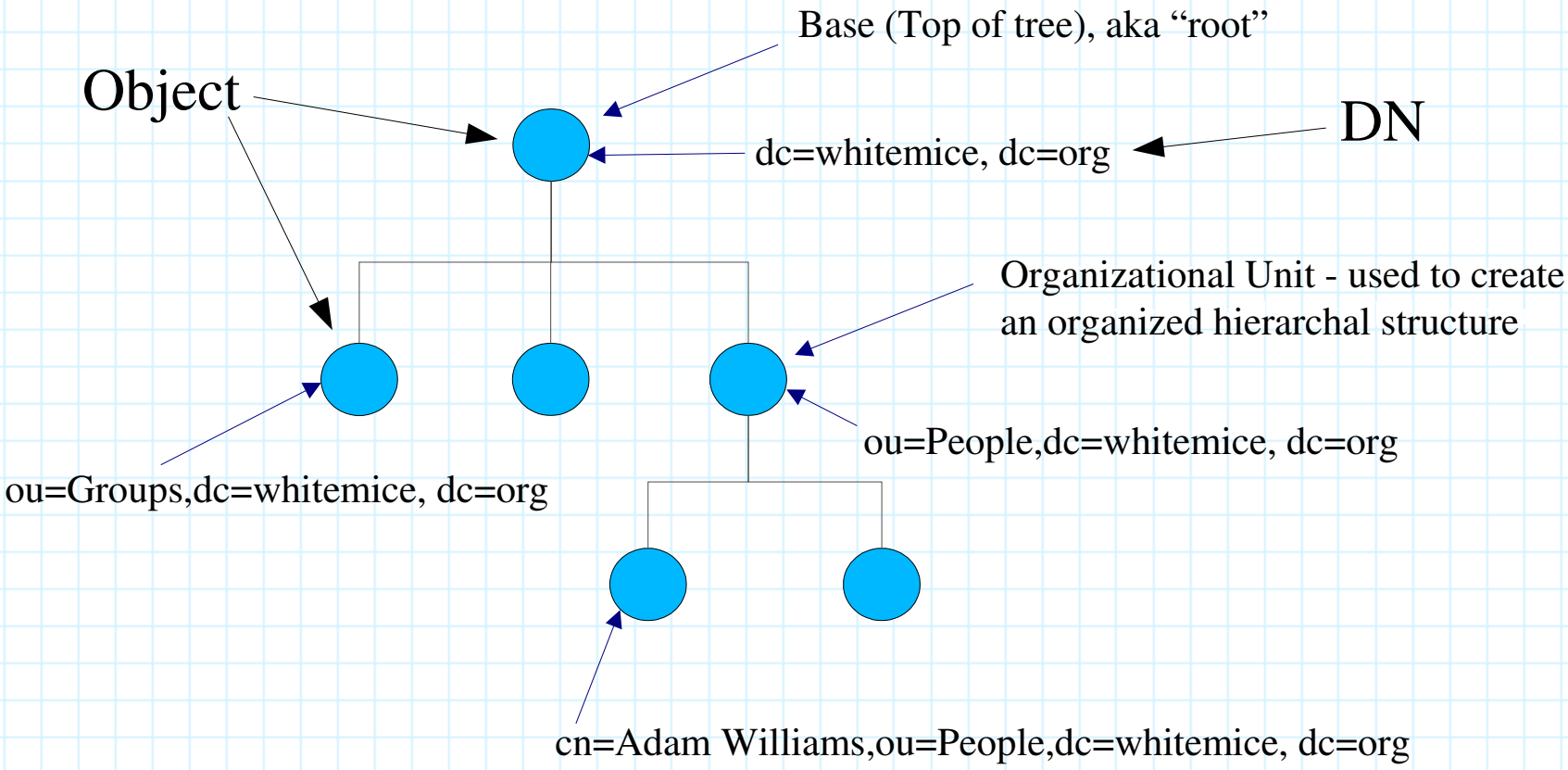
- Naming
  - The structure of the directory
    - What the 'name' of the directory is.
      - Where given content will be located in the directory.
- Topology
  - How the DSAs are configured.
  - How portions of the directory are distributed on the network.
- Schema
  - Rules that control what content cane be contained in the directory.
- Data
  - What content will be stored in the directory

# Tree Naming

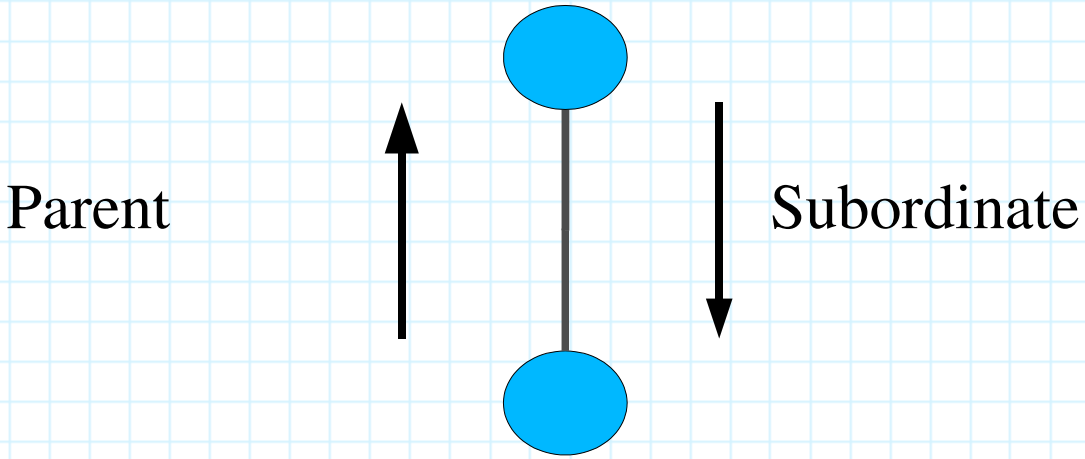
- A Dit must have a root, various conventions exist for naming the base of a Dit.
  - X.500
    - The 'original' style
    - Looks like: o=Foo,c=US
  - RFC2247
    - Looks like: dc=foo,dc=com
    - Recommended!
      - Allows easy integration with DNS, and server auto location.
      - Used by M$-Active Directory
  - Just dumb
    - Look like: dc=foo.com

# The Dit

## (Directory information tree)



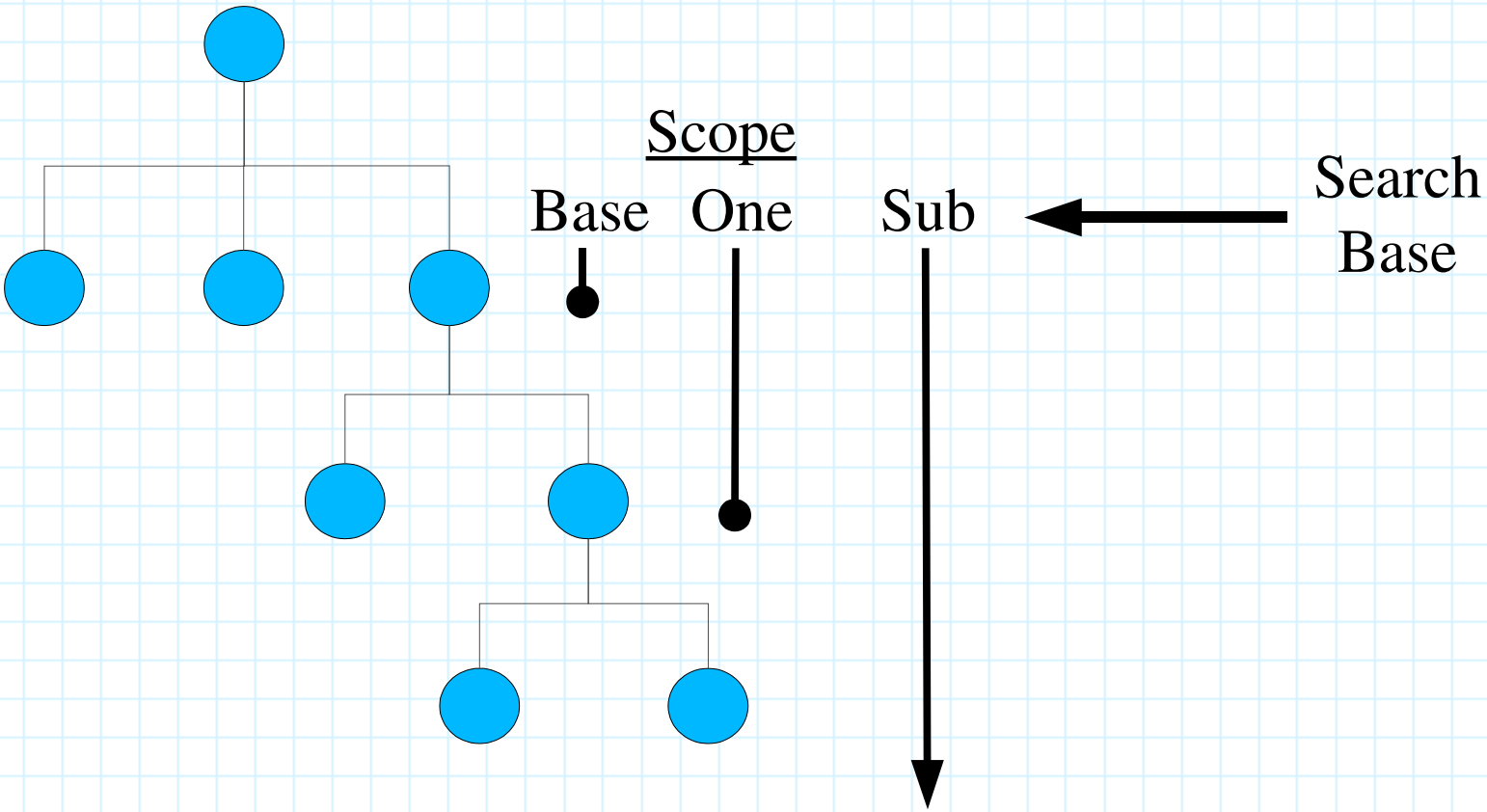
# Subordinates




- The operational attribute “hasSuborinates” can be used to determine if an object has subordinates. (OpenLDAP)

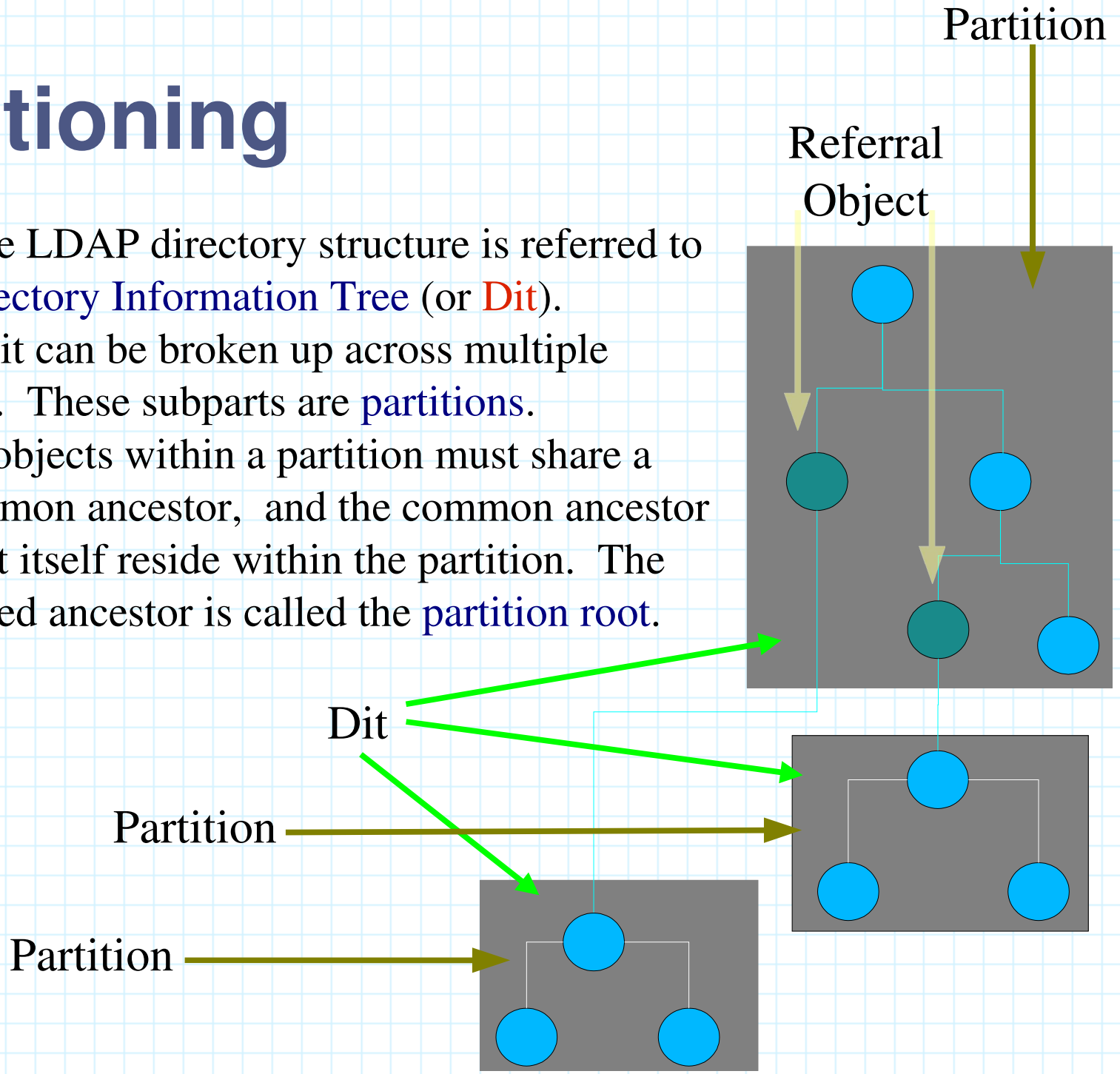
# Search Directives

•Queries have a base (starting point) and a scope.



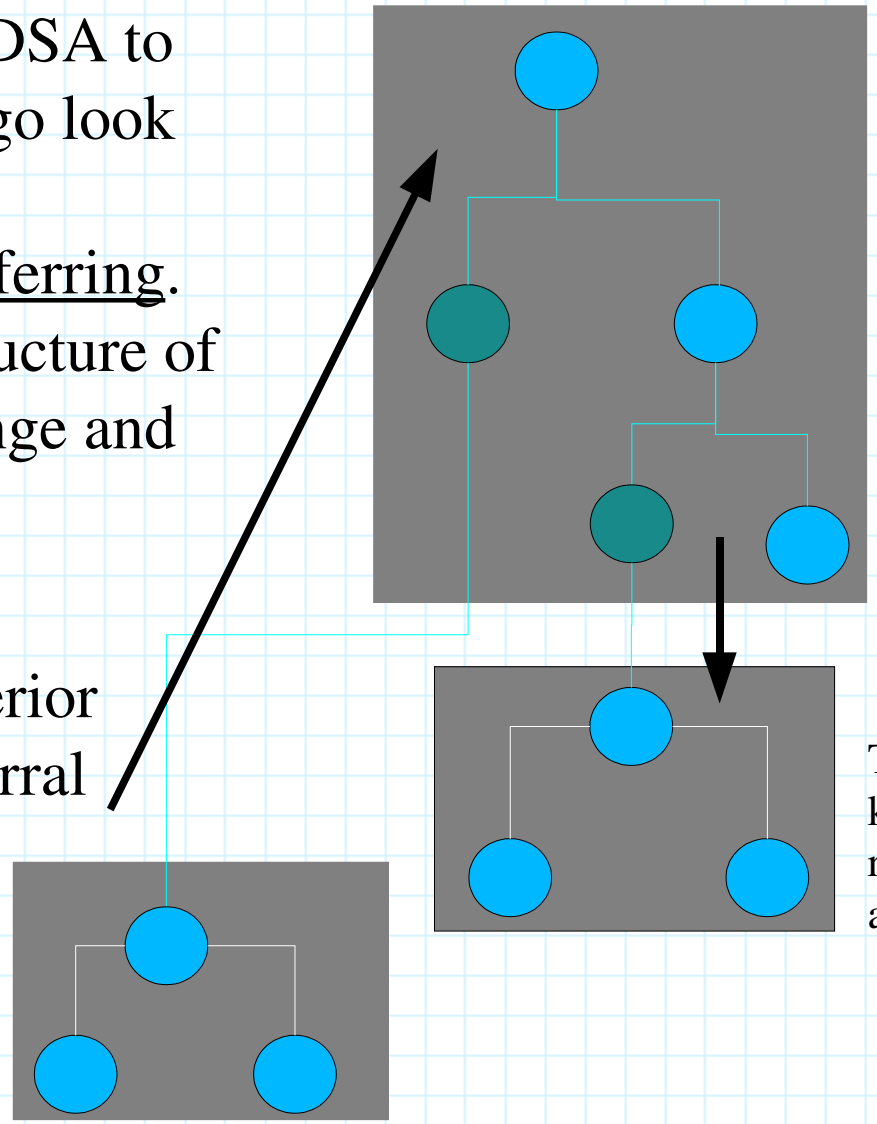
# Partitioning

- The entire LDAP directory structure is referred to as the Directory Information Tree (or Dit).
  - The Dit can be broken up across multiple DSAs. These subparts are partitions.
    - All objects within a partition must share a common ancestor, and the common ancestor must itself reside within the partition. The shared ancestor is called the partition root.



# Refering

- The LDAP protocol provides a mechanism for a DSA to instruct a client to 'go look elsewhere'.
  - This is called referring.
    - Allows the structure of the Dit to change and grow.



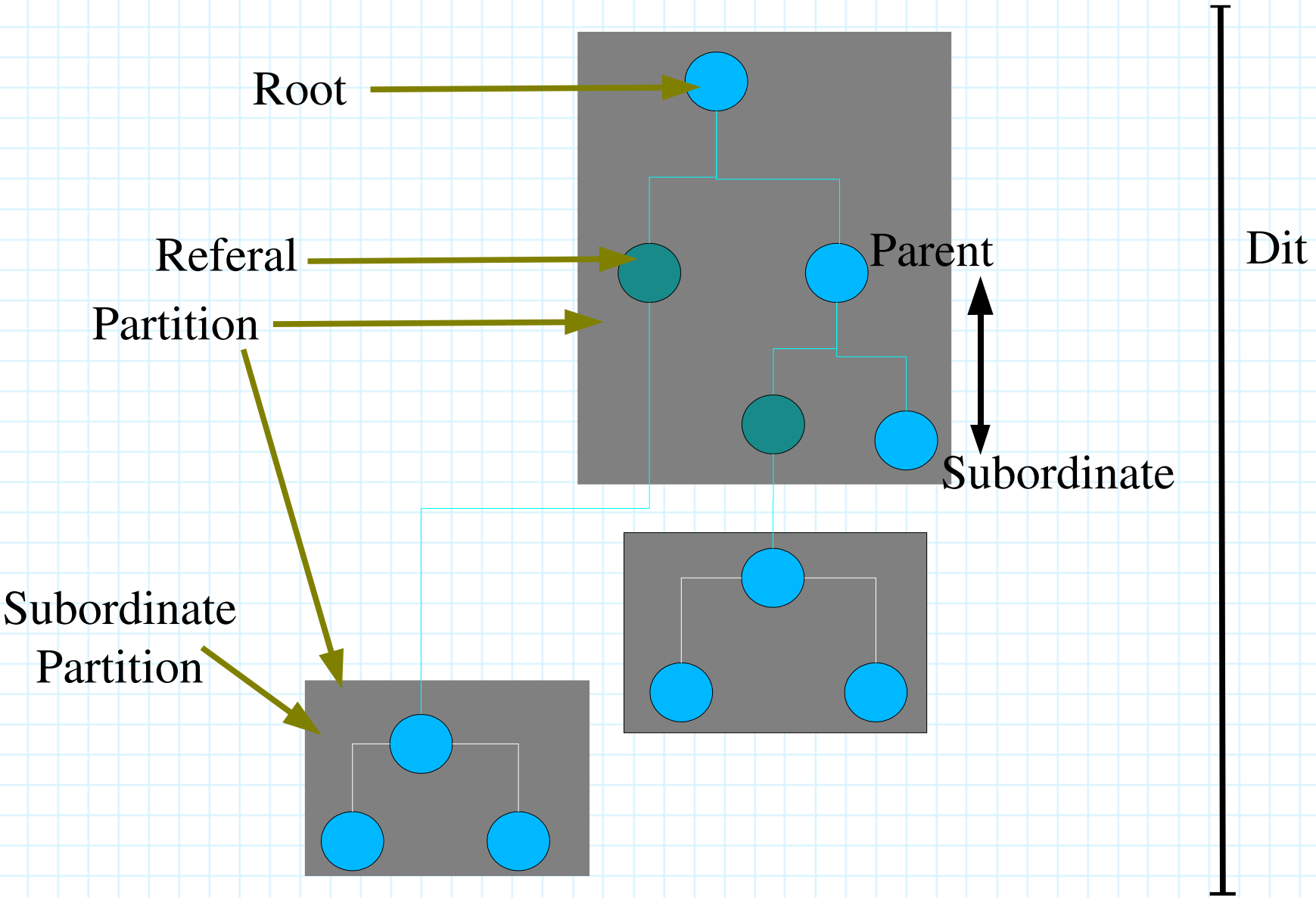
Subordinate Referral

The DSA queried knows the scope requested exists on another partition.

Superior Referral

The DSA queried does not contain the scope requested, client is sent 'upstream'.

# Dit Overview



# An Object

dn: cn=PC02019,ou=CIFS Devices,ou=Devices,o=Morrison Industries,c=US
objectClass: top
objectClass: device
objectClass: ipHost
objectClass: morrisonworkstation
cn: PC02019
ipHostNumber: 192.168.1.215
l: Grand Rapids
seeAlso: ou=Grand Rapids,ou=Sites,o=Morrison Industries,c=US
seeAlso: relativeDomainName=PC02019,zoneName=morrison.iserv.net,...
seeAlso: relativeDomainName=215.1,zoneName=168.192.in-addr.arpa,,,,
seeAlso: uid=PC02019$,ou=System Accounts,o=Morrison Industries,c=US
morrisonttystation: CIS

Multi-Valued

Single Valued

Attribute

Value

# Object Components

DN

RDN | Container

dn: cn=PC02019,ou=CIFS Devices,ou=Devices,o=Morrison Industries,c=US

objectClass: top
objectClass: device
objectClass: ipHost
objectClass: morrisonworkstation

objectclasses

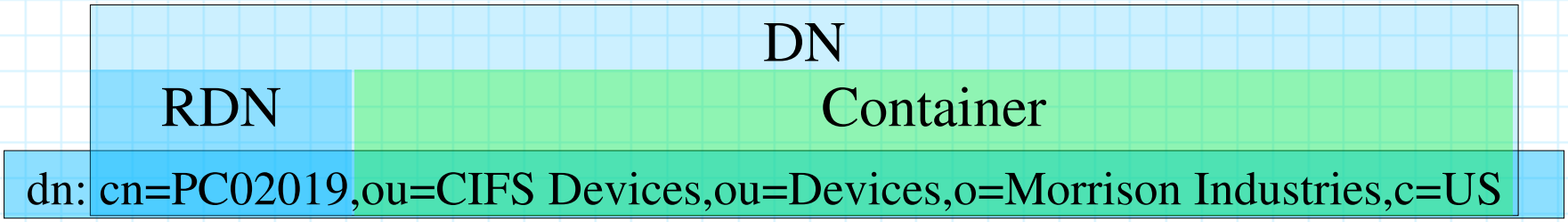
cn: PC02019
ipHostNumber: 192.168.1.215
l: Grand Rapids
seeAlso: ou=Grand Rapids,ou=Sites,o=Morrison Industries,c=US
seeAlso: relativeDomainName=PC02019,zoneName=morrison.iserv.net,....
seeAlso: relativeDomainName=215.1,zoneName=168.192.in-.....
seeAlso: uid=PC02019$,ou=System Accounts,o=Morrison Industries,c=US
morrisonttystation: CIS

data

hasSubordinates: FALSE
entryUUID: a0a5e53a-9bb2-1028-9db4-f4db6d75e42a
modifyTimeStamp: 20041006211314Z

meta-data

# DN




- The parent object acts as a container for subordinate objects.
- Makes the object unique within the container.
  - This is an attribute value pair that exists within the object's data.
- The distinguished name uniquely identifies an object with the Dit.
  - DN changes are generally frowned upon, thus the RDN should be a stable value.

# Multi-Valued RDNs

RDN

```
dn: cn=postgres+ipServiceProtocol=tcp,ou=IpServices,ou=NSS,....
objectClass: ipService
objectClass: top
ipServicePort: 5432
ipServiceProtocol: tcp
cn: postgres
```

- If one attribute cannot uniquely qualify an object within a container, multiple attribute value pairs may be used as the RDN.
  - Attribute value pairs are delimited with a plus sign.

# objectClass

| |
|---|
| dn: cn=postgres+ipServiceProtocol=tcp,ou=IpServices,ou=NSS,.... |
| objectClass: ipService<br>objectClass: top |
| ipServicePort: 5432<br>ipServiceProtocol: tcp<br>cn: postgres |

- The objectclass declares the “type” of an object:
  - Person, Device, Service, Building, etc...
- Every object contains an objectclass “top”

# objectClass declaration

```
dn: cn=postgres+ipServiceProtocol=tcp,ou=IpServices,ou=NSS,....
objectClass: ipService
objectClass: top
ipServicePort: 5432
ipServiceProtocol: tcp
cn: postgres
```

```
objectclass ( 1.3.6.1.1.1.2.3 NAME 'ipService' SUP top STRUCTURAL
        DESC 'Abstraction an Internet Protocol service'
        MUST ( cn $ ipServicePort $ ipServiceProtocol )
        MAY ( description ) )
```

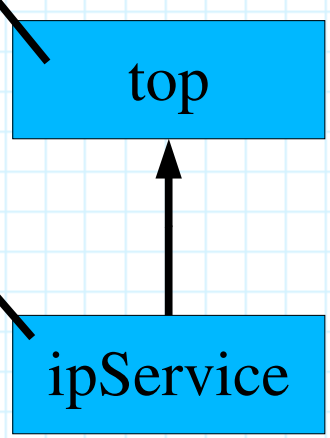
- Objectclasses are declared in the schema of the DSA.
  - What attributes are required, what attributes are allowed.
  - Description
  - Name

# Objectclass Inheritance

```
dn: cn=postgres+ipServiceProtocol=tcp,ou=IpServices,ou=NSS,....
objectClass: ipService
objectClass: top
ipServicePort: 5432
ipServiceProtocol: tcp
cn: postgres
```

```
objectclass ( 1.3.6.1.1.1.2.3 NAME 'ipService' SUP top STRUCTURAL
        DESC 'Abstraction an Internet Protocol service'
        MUST ( cn $ ipServicePort $ ipServiceProtocol )
        MAY ( description ) )
```

- Objectclasses inherit properties from their 'superiors' just like objects in traditional OO systems.
  - All attributes are public in an LDAP objectclass.



# objectClass Type

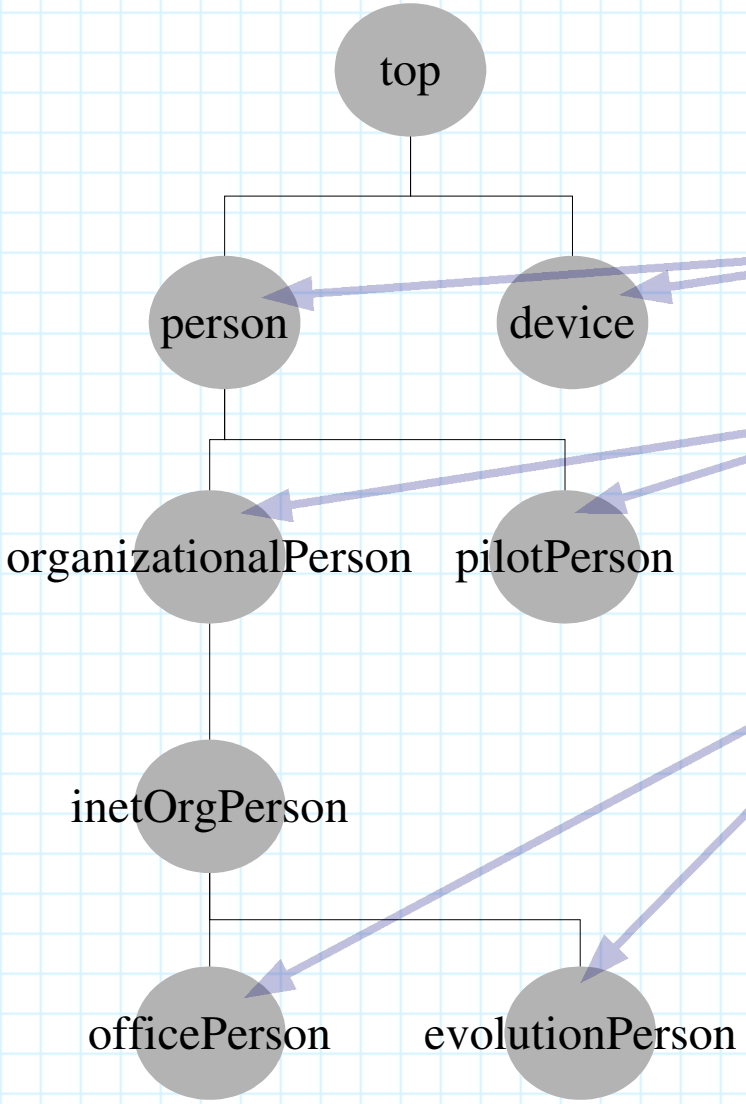
```
dn: cn=postgres+ipServiceProtocol=tcp,ou=IpServices,ou=NSS,....
objectClass: ipService
objectClass: top
ipServicePort: 5432
ipServiceProtocol: tcp
cn: postgres
```

```
objectclass ( 1.3.6.1.1.1.2.3 NAME 'ipService' SUP top STRUCTURAL
        DESC 'Abstraction an Internet Protocol service'
        MUST ( cn $ ipServicePort $ ipServiceProtocol )
        MAY ( description ) )
```

AUXILIARY
or
STRUCTURAL

- An objectclass is either
  - Structural
    - Declares a type: account, person, etc..
  - Auxiliary
    - Declares additional information

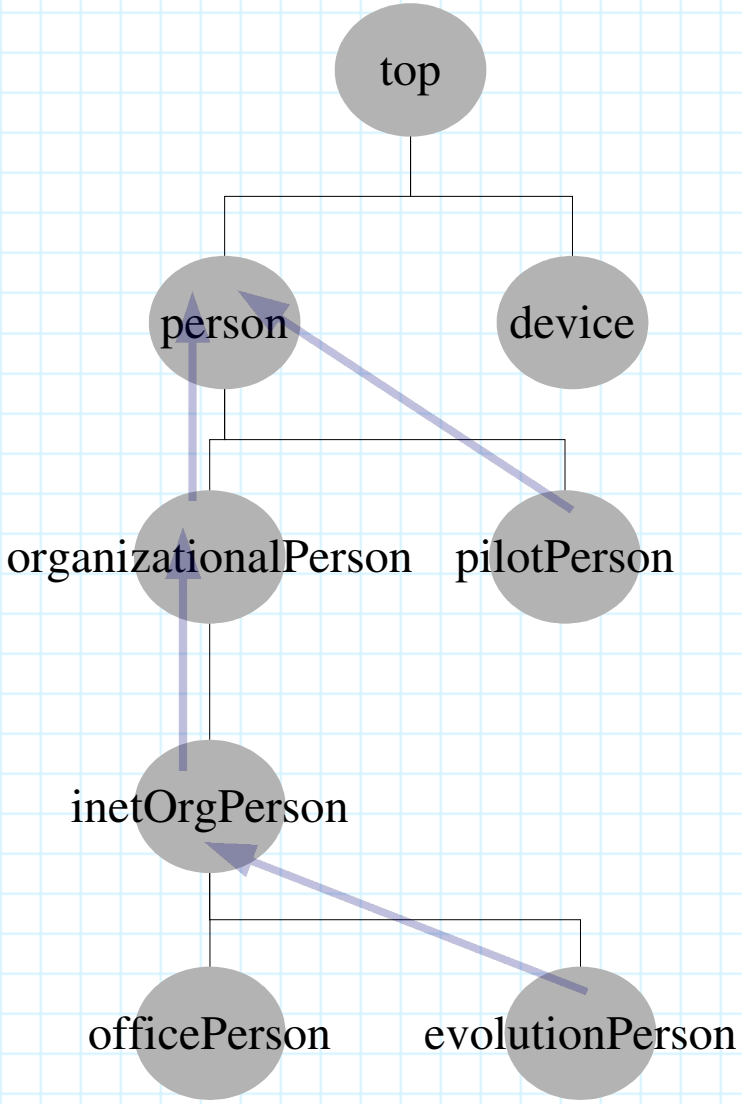
# The Chain




- An object can only contain a single chain of structural objectclasses; this is because a structural objectclass declares type.
  - An object cannot be both a device and a person.
  - An object cannot be both an inetOrgPerson and a pilotPerson.
  - An object cannot be both an officePerson and a evolutionPerson.

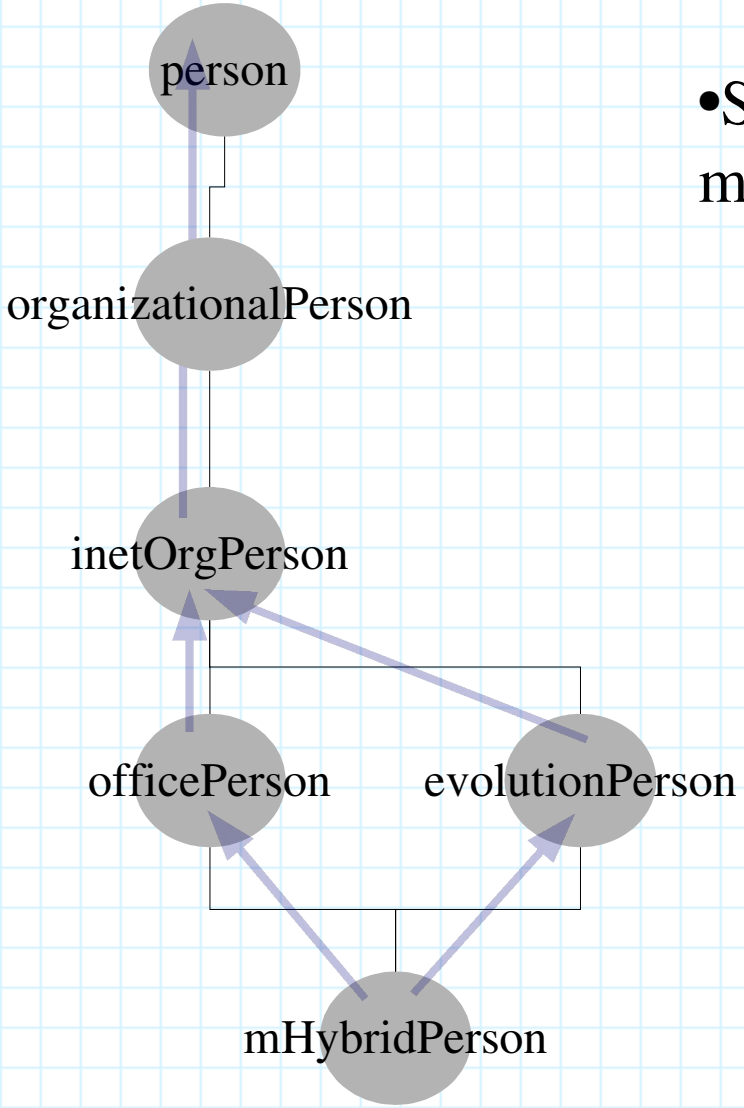
This is called the:
Structural Objectclass Chain

# Inheritance Chain




- Objectclass membership is inherited just like schema attribution is inherited.
  - A pilotPerson is a person.
  - An evolutionPerson is an inetOrgPerson, an organizationalPerson, and a person.

# Multiple Inheritence



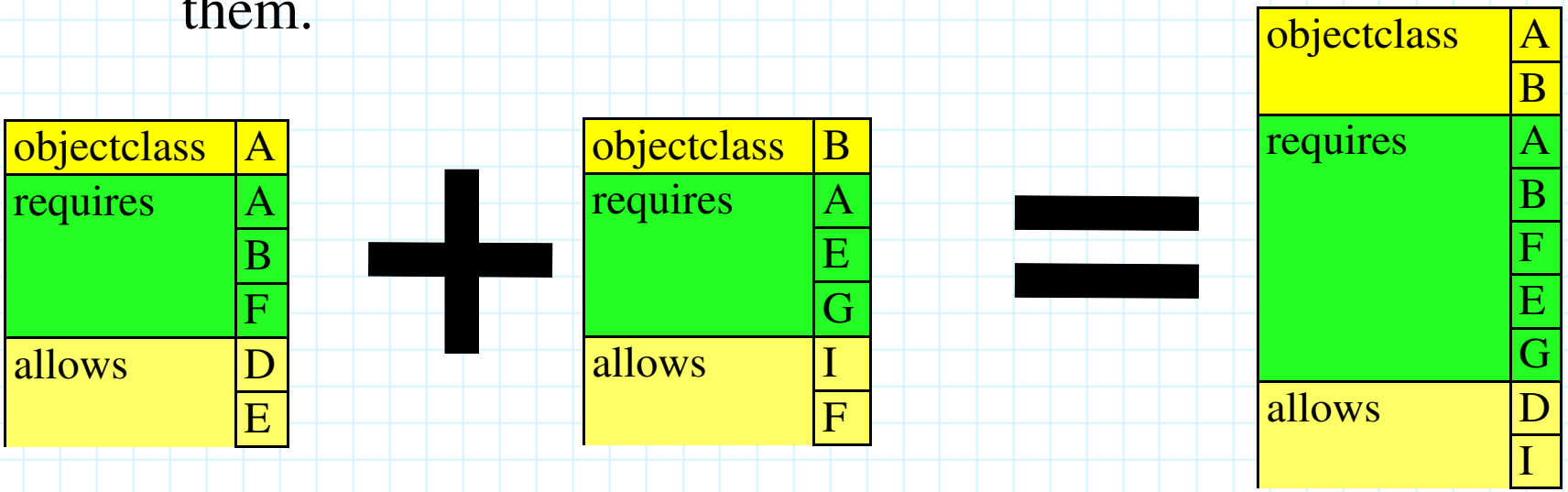

- Structural objectclasses support multiple inhertance.
  - Divergent structural objectclass chains can be healed.
    - Multiple inheritance should be used carefully or it can distort the data model.
      - For instance, you would not descend from both device and person.

# mHybridPerson

objectclass ( 1.3.6.1.4.1.6921.1.12
NAME 'mHybridPerson'
DESC 'Combine objectclasses to support multiple MUAs'
SUP ( officePerson $ evolutionPerson )
STRUCTURAL )

# Attribute Aggregation

- The attributes required or allowed by objectclasses aggregate when combined.
  - Permitted attributes are escalated to required, when one or more objectclass combined requires them.



# Aggregation Example

```
objectclass ( 2.5.6.14 NAME 'device'
        DESC 'RFC2256: a device'
        SUP top STRUCTURAL
        MUST cn
        MAY ( serialNumber $ seeAlso $ owner $ ou $ o $ l $ description ) )

objectclass ( 1.3.6.1.1.1.2.6 NAME 'ipHost' SUP top AUXILIARY
        DESC 'Abstraction of a host, an IP device'
        MUST ( cn $ ipHostNumber )
        MAY ( l $ description $ manager ) )

objectclass ( 1.3.6.1.4.1.6921.1.16
    NAME 'morrisonworkstation'
      DESC 'Site Specific Workstation Attributes'
    SUP device STRUCTURAL
      MAY ( morrisonttystation $ netbiosservice ) )
```

Required:

- cn
- ipHostNumber

Allowed

- serialNumber
- description
- manager
- netbiosservice
- etc...

# OID

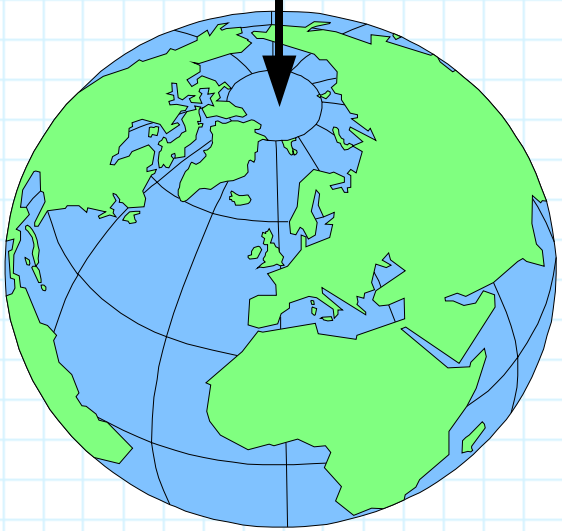
- The definition of every element of LDAP schema, attributes and objects, contains a globally unique identifier.
  - You CANNOT make up your own OID or reuse anyone elses.
  - You MUST register for an OID in order to create LDAP schema.

    http://www.iana.org/cgi-bin/enterprise.pl

```
objectclass ( 1.3.6.1.4.1.6921.1.16
    NAME 'morrisonworkstation'
      DESC 'Site Specific Workstation Attributes'
    SUP device STRUCTURAL
      MAY ( morrisonttystation $ netbiosservice ) )
```

OID

By “globally” we mean “globally”.



http://asn1.elibel.tm.fr/cgi-bin/oid/display

# Reading OIDs

- 1.3 :
  - 1.3.6 : US Department Of Defense
    - 1.3.6.1 : Internet
      - 1.3.6.1.1 : Directory
        - 1.3.6.1.1.1 : NIS Schema (RFC2307)
      - 1.3.6.1.4 : Private
        - 1.3.6.1.4.1 : Enterprises
          - 1.3.6.1.4.1.2 : IBM
          - 1.3.6.1.4.1.9 : Cisco
          - 1.3.6.1.4.1.6921 : Morrison Industries
            - 1.3.6.1.4.1.6921.1 : *attributes*
              - 1.3.6.1.4.1.6921.1.24 : morrisonvpnaccess
            - 1.3.6.1.4.1.6921.2 : *objectclasses*
              - 1.3.6.1.4.1.6921.2.10 : morrisonsite
          - 1.3.6.1.4.1.7165 : The Samba Project

# Special Objectclasses

referral instructs clients to continue their search elsewhere.

```
dn: ou=Metadirectory,dc=Whitemice,dc=Org
objectClass: referral
objectClass: extensibleObject
dc: subtree
ou: Metadirectory
ref: ldap://ldap.whitemice.org/ou=Metadirectory,dc=Whitemice,dc=Org/
```

extensibleObject permits any collection of attributes known to the DSA.

# Special Objectclasses

- The "alias" objectclass is the 'symbolic link' of the directory world.
  - aliasObjectName directs the DSA to check an alternative object.
    - Aliases can extract a serious performance penalty and should be used when no other remedy exists.

```
dn: cn=fred,ou=People,dc=Linux,dc=net
objectClass: top
objectClass: alias
objectClass: extensibleObject
aliasedObjectName: uid\=george\,ou\=People\,dc\=Linux\,dc\=net
cn: fred
```

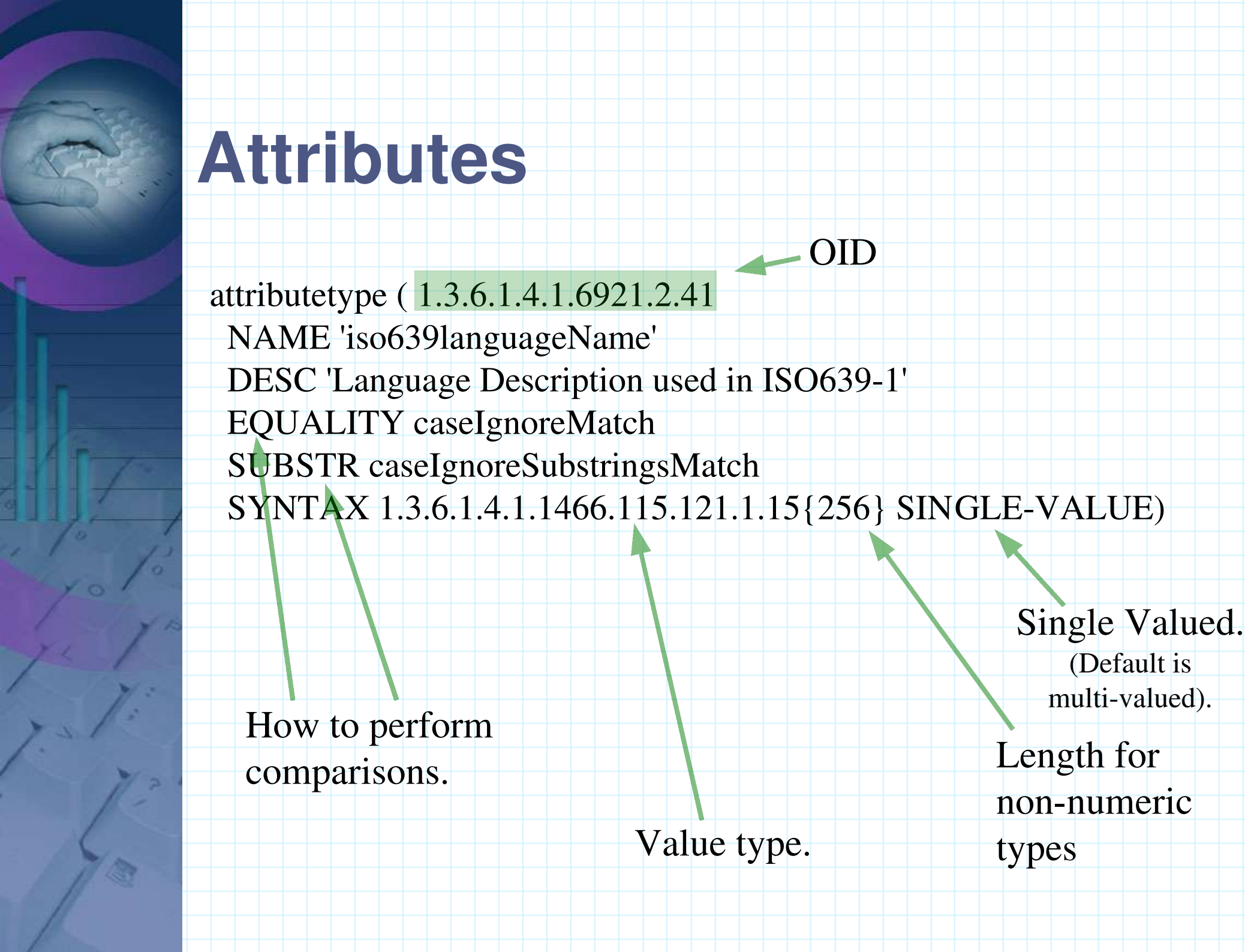
# The rootDSE

dn:
namingContexts: dc=morrison-ind,dc=com
namingContexts: dc=cisco-inc,dc=com
namingContexts: dc=mor-value,dc=com
namingContexts: dc=mormail,dc=com
namingContexts: dc=traidservice,dc=com
namingContexts: o=Morrison Industries,c=US
namingContexts: o=localfiles
supportedControl: 2.16.840.1.113730.3.4.2
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.1466.20037
supportedLDAPVersion: 2
supportedLDAPVersion: 3
supportedSASLMechanisms: GSSAPI
subschemaSubentry: cn=Subschema

- Every DSA contains a rootDSE object.
  - This object allows clients to discover information about the DSA:
    - The Dits and partitions hosted
    - Features and extensions supported
    - How to download the supported schema

OpenLDAP command used to retrieve the rootDSE: ldapsearch -x -b '' -s base '(objectclass=*)' '+'

# Attributes


OID
attributetype ( 1.3.6.1.4.1.6921.2.41
NAME 'iso639languageName'
DESC 'Language Description used in ISO639-1'
EQUALITY caseIgnoreMatch
SUBSTR caseIgnoreSubstringsMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} SINGLE-VALUE)
How to perform comparisons.
Value type.
Length for non-numeric types
Single Valued.
(Default is multi-valued).

# Attribute Types

| Data Type | OID | Description |
|---|---|---|
| Binary | 1.3.6.1.4.1.1466.115.121.1.5 | BER/DER data |
| Boolean | 1.3.6.1.4.1.1466.115.121.1.7 | boolean value |
| Distinguished Name | 1.3.6.1.4.1.1466.115.121.1.12 | DN |
| Directory String | 1.3.6.1.4.1.1466.115.121.1.15 | UTF-8 string |
| IA5String | 1.3.6.1.4.1.1466.115.121.1.26 | ASCII string |
| Integer | 1.3.6.1.4.1.1466.115.121.1.27 | Integer |
| Name and Optional UID | 1.3.6.1.4.1.1466.115.121.1.34 | DN plus UID |
| Numeric String | 1.3.6.1.4.1.1466.115.121.1.36 | Numeric String |
| OID | 1.3.6.1.4.1.1466.115.121.1.38 | Object Identifier |
| Octet String | 1.3.6.1.4.1.1466.115.121.1.40 | Arbitrary Octets |
| Printable String | 1.3.6.1.4.1.1466.115.121.1.44 | Printable String |

# Operational Attributes

- structuralObjectClass
  - The objectclass of the bottom of the objects structural objectclass chain.
- creatorsName
  - Identity of the entity that created to object
- createTimestamp
  - When the object was created
  - Example: 20040320225722Z
- entryUUID
  - Globally unique identifier of the object.
  - Example: a0a5e53a-9bb2-1028-9db4-f4db6d75e42a
- entryCSN
  - Context timestamp, used for replication, syncronization.
  - Example: 20041006213314Z#000001#00#000000

# Operational Attributes

- modifiersName
  - Identity of the last entity to modify the object's data
- modifyTimestamp
  - Last time an object's data was modified.
  - Example: 20041006213314Z
- subschemaSubentry
  - Where in the Dit the client application can find information about the schema supported by the DSA.
  - Example: cn=Subschema
- hasSubordinates
  - Whether or not the object has subordinate object.

# Attribute Overview

- An attribute has
  - An OID
  - A Name
    - Possibly multiple aliases
  - A Syntax (Data Type)
    - Appropriate matching rules
  - Is Multi-valued unless specified as Single Valued
  - Must be included into an objectclass definition
    - Required
    - Allowed
  - An instance must contain a value; no NULLs.

# Operations

- Interacting with a DSA is, at least theoretically, simple.
- Operations:
  - Connect
    - Bind : Authenticate a connection to the DSA.
      - This may include the use of SASL, Kerberos, etc...
      - Search : Locate objects matching some criteria.
      - Compare : Return True or False to some query.
      - Add : Create a new object in the Dit.
      - Delete : Remove an object from the Dit.
      - Modify : Change the attribute values of an object.
      - Rename : Change the DN of an object.
    - Unbind : Disconnect from the DSA.

# Extended Operations

- Extended operations and controls provide a mechanism for additional (possibly vendor specific) functionality to be added to the DSA

```
dn:
namingContexts: dc=morrison-ind,dc=com
namingContexts: dc=cisco-inc,dc=com
namingContexts: dc=mor-value,dc=com
namingContexts: dc=mormail,dc=com
namingContexts: dc=traidservice,dc=com
namingContexts: o=Morrison Industries,c=US
namingContexts: o=localfiles
supportedControl: 2.16.840.1.113730.3.4.2
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.1466.20037
supportedLDAPVersion: 2
supportedLDAPVersion: 3
supportedSASLMechanisms: GSSAPI
subschemaSubentry: cn=Subschema
```

- Clients can discover what controls or extensions a DSA supports by querying its rootDSE.
- Contsols and extensions are assigned OIDs, like objectclasses and attributes, so that they can be uniquely identified.

# Some Controls & Extensions

- Control: ManageDsaIT
  - OID: 2.16.840.1.113730.3.4.2
    - Disables chasing referrals, allows administration of partitioning.
- Extension: Password Modify
  - OID: 1.3.6.1.4.1.4203.1.11.1
    - Used to request the DSA change a password attribute, this allows the DSA to enforce password policies, perform the encryption, and possibly integrate with services such as Kerberos.
- Extension: Start TLS
  - OID: 1.3.6.1.4.1.1466.20037
    - The StartTLS extension requests that the DSA use encryption (SSL) on the current connection.

# About LDIF

- LDAP Directory Information File
  - How data is exported from or imported into a Dit.
  - A text file.
    - Lines in an LDIF file are limited to a length of 78 characters.
    - Lines beginning with a space are assumed to be a continuation of the previous line.
  - Every DSA (?) provides tools for managing LDIF files.
    - Even M$-Active Directory provides command line tools for loading from and unloading to LDIF.
  - Objects begin with their DN
  - Objects are terminated by a blank line

```
dn: ou=Brighton,ou=Sites,o=Morrison Industries,c=US
objectClass: top
objectClass: organizationalUnit
objectClass: domainRelatedObject
objectClass: morrisonsite
ou: Brighton
telephoneNumber: 8102276311
facsimileTelephoneNumber: 8102276868
street: Old US 23
postalCode: 48116
postalAddress: 1183 Old US 23$Brighton, Mi. 48116
physicalDeliveryOfficeName: Brighton
l: Brighton
associatedDomain: morrison-ind.com
morrisonemaildrop: internet.orders:shared+internet.morrison.order.brt@morrison
 -ind.com
morrisonbranch: BRT
st: Michigan
morrisonphoneextension: 1700
morrisonmtabranchcode: 0107
morrisonspeeddialcode: 63
opengroupwaregrojectid: P116290
morrisonservicewages: 1000
```

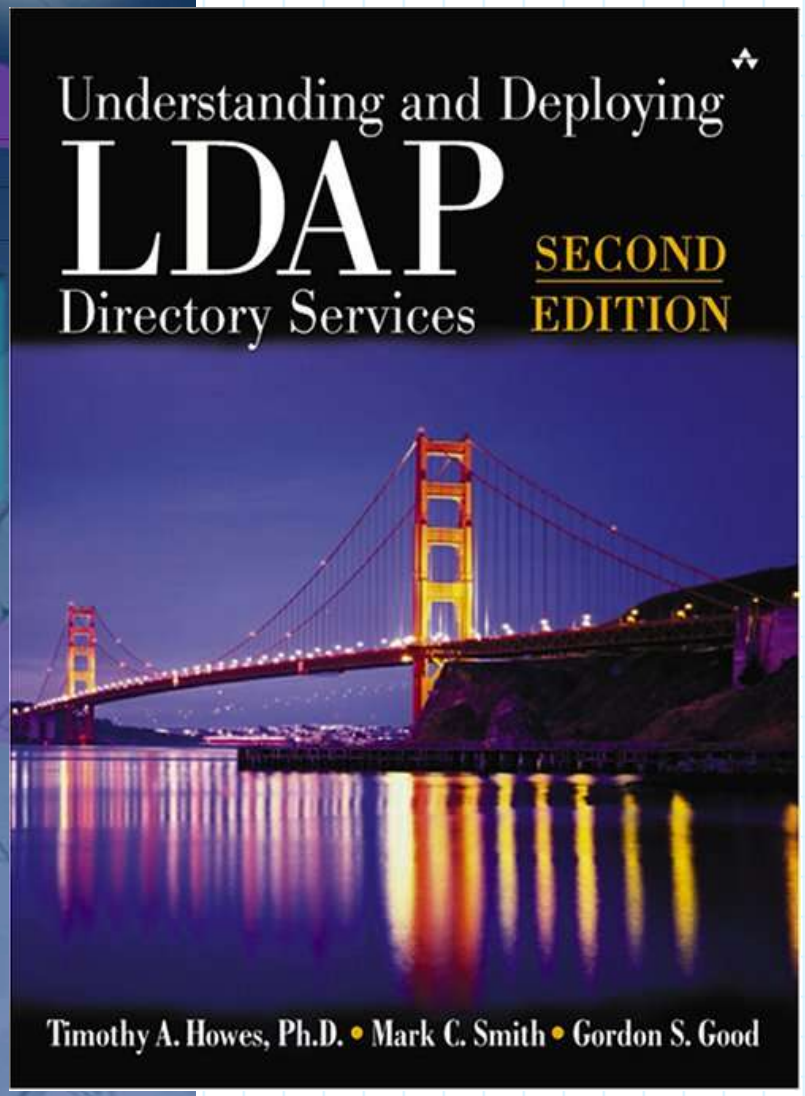
DN specified the beginning of an object specification.

Well formed LDIF list objectClass values before data values.

A wrapped line!

Object terminated with a blank line.

# The Book




**Understand and Deploying**

**LDAP**

**Directory Services** (2nd Edition)

(May 2, 2003)

AKA: "The Bible"

Hardcover: 936 pages

Dimensions: 1.86 x 9.52 x 7.68

Publisher: Addison-Wesley Pub Co

ISBN: 0672323168