Naming in Distributed Systems

Sistemas Distribuídos

Chapter 9, Livro do George Coulouris

Names

- **Names**: Identification of objects

  - resource sharing: Internet domain names

  - communication: domain name part of email address

Name Services vs Directory Services

- **Name Services**: entries of the form <name, attributes>, where attributes are typically network addresses.
- Type of lookup queries:
  - name → attribute values
  - also called *name resolution*

- **Directory Services**
- <name, attributes> entries
- Types of lookup queries:
  - name → attribute values
  - attribute values → names

Name and Directory Services

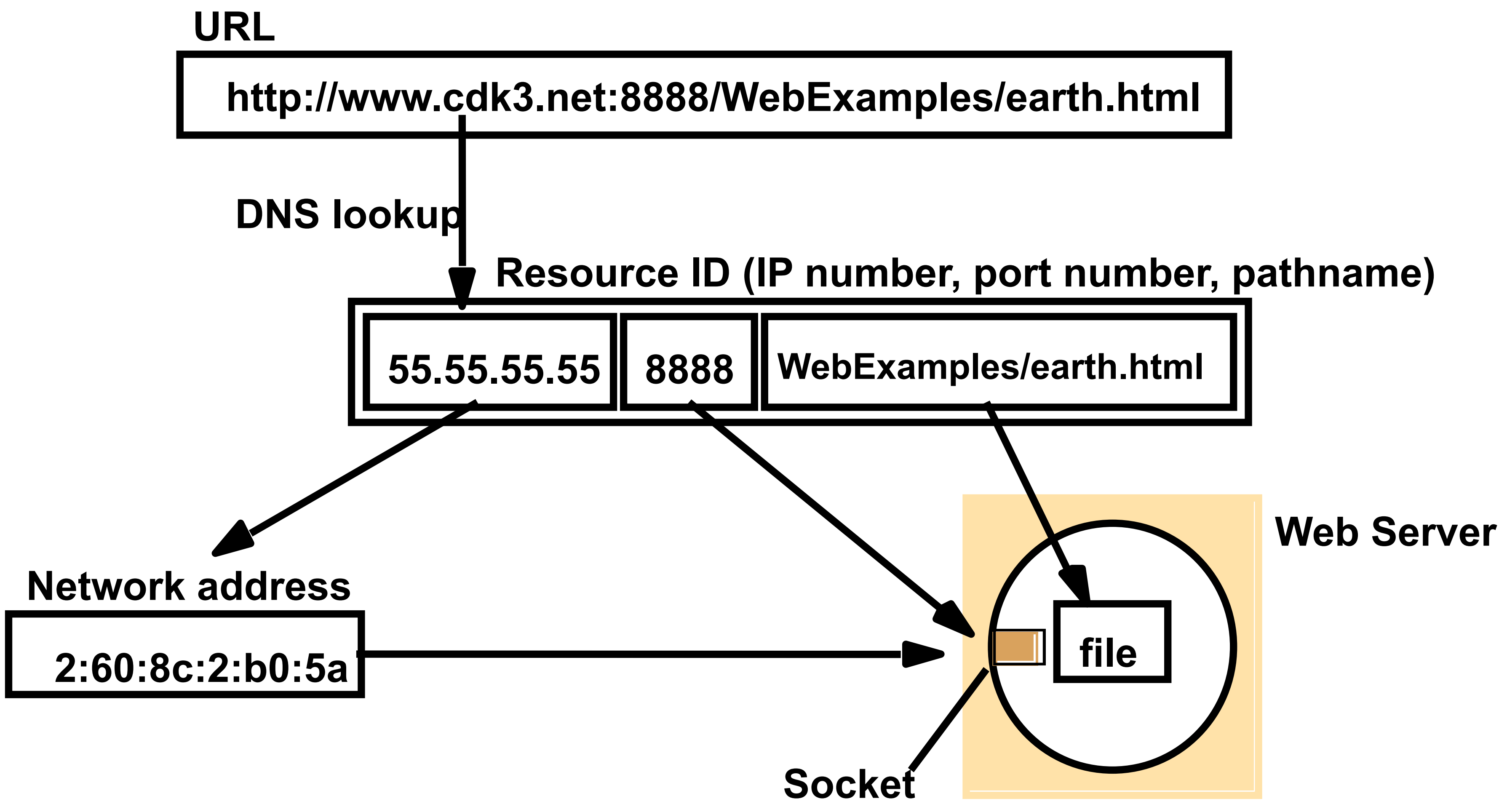
- **DNS**: maps domain names to the attributes of a host computer (including IP address). Name service used accross the Internet.
- **X.500 Directory Service**: can be used to map the name of person into a set of attributes (email address, telephone number, ...). Standardized directory service.
- **RMI Registry/CORBA Naming Service**: maps the name of a remote object into its remote reference.
- **LDAP**: directory service, lightweight implementation of X.500. Used in intranet applications.
- **JINI:** has a discovery service for spontaneous networks.
- **UDDI**: directory service for Web-Services.

Uniform Resource Identifiers (URIs)

- **Uniform Resource Locators (URL):** URLs are essentially addresses of Web resources.
- They have a small drawback: if a resource is deleted or if it moves there will be dangling links to the resource containing the old URL.

- **Uniform Resource Names (URN):** URNs are intended to solve that problem and to provide richer modes of finding resources on the Web. A URN of a Web resource will persist even when that resource moves. The owner of the resource registers its name, along its current URL. The owner has to register the new URL if the resource moves.

URL Mapping


URL
http://www.cdk3.net:8888/WebExamples/earth.html
DNS lookup
Resource ID (IP number, port number, pathname)
55.55.55.55
8888
WebExamples/earth.html
Network address
2:60:8c:2:b0:5a
Web Server
file
Socket

Name Servers

- A Name Service like DNS stores a large database and is used by a large population.
- The information is not stored on a single server.
- DNS uses database partitioning, data replication and caching, to improve performance, scalability and availability.

Data Management in Name Servers

- **Partitioning**

  - no one name server can hold all name and attribute entries for entire network, in particular in Internet

  - name server data partitioned according to domain
- **Replication**

  - a domain has usually more than one name server

  - availability and performance are enhanced
- **Caching**

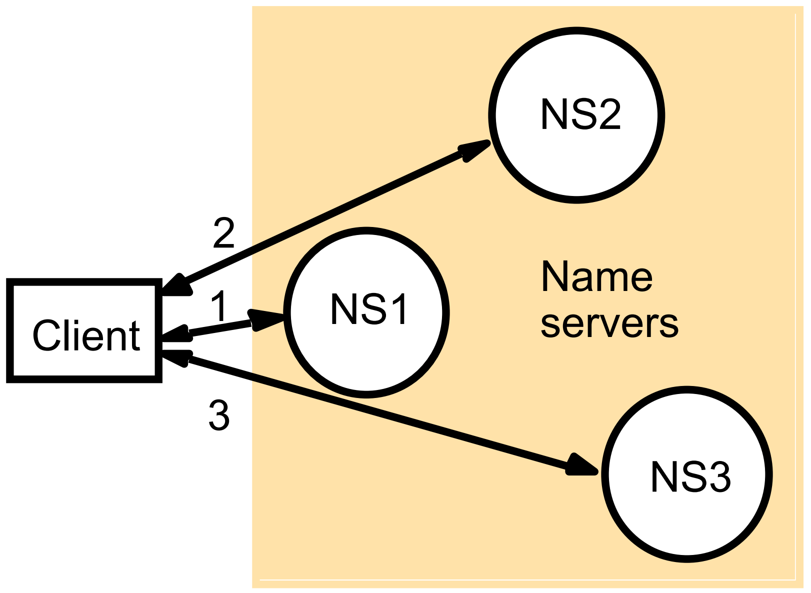
  - servers may cache name resolutions performed on other servers

  - client lookup software may equally cache results of previous requests

Name Servers and Navigation

- Interactive Navigation
- Server-Enabled Navigation:
  - non-recursive
  - recursive
- Multicast Navigation

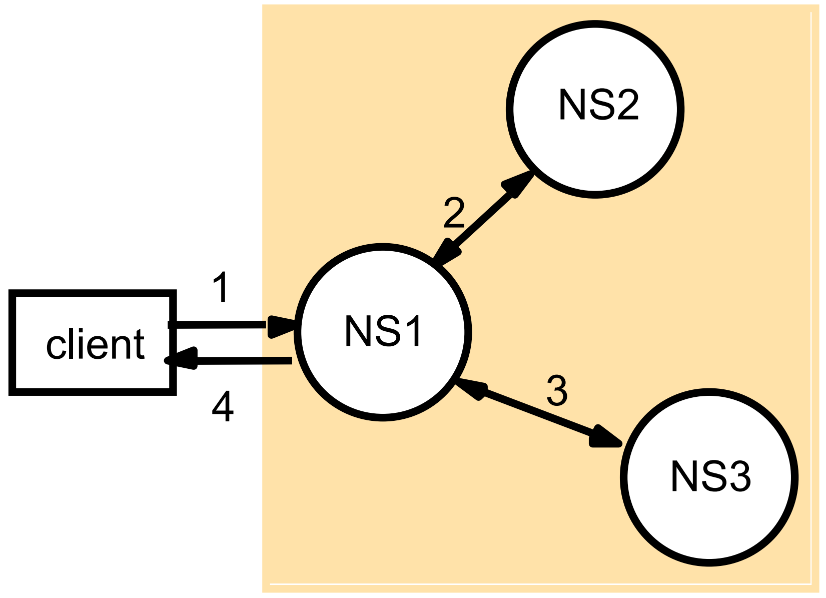
Iterative Navigation



A client iteratively contacts name servers NS1–NS3 in order to resolve a name.

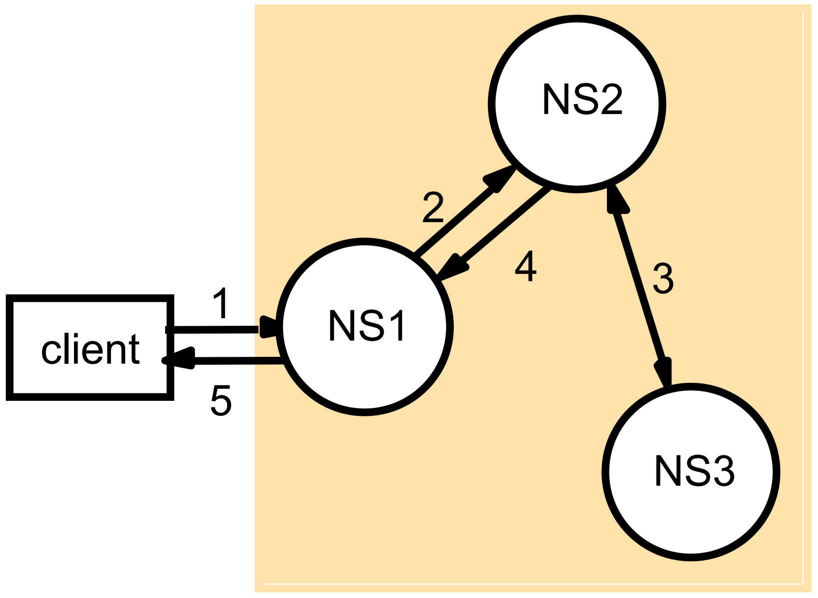
- used in DNS
- client contacts one NS
- NS either resolves name, or suggests other name server to contact
- resolution continues until name resolved or name found to be unbound

Server-Controlled Navigation



Non-recursive server-controlled

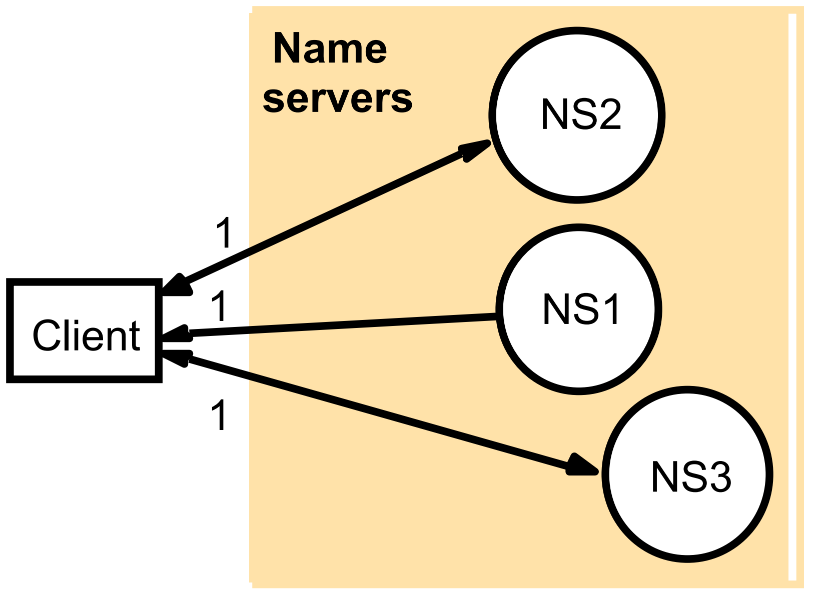
– server contacts peers if it cannot resolve name itself by broadcast or iteratively by direct contact.



Recursive server-controlled

– if name cannot be resolved, server contacts superior server responsible for a larger prefix of the name space.
– recursively applied until name resolved.
– can be used when clients and low-level servers are not entitled to directly contact high-level servers.

Multicast Navigation



Only the server that holds the named attributes respond to the request.
When a required name is unbound one of the servers can take care of that request outside the group.

Domain Name System: DNS

- DNS was presented in 1987
- Before that: Internet Naming Scheme.

  All host names and addresses were held in a single central master file and downloaded by FTP to the computers that required it.

  Problems: did not scale and did not allow for local organizations to administer their own name contexts.
- DNS mainly stores computer name <->IP address mappings

  - domains = naming domains

  - top-level/organizational domains:

  `com, edu, net, us, de, ...`

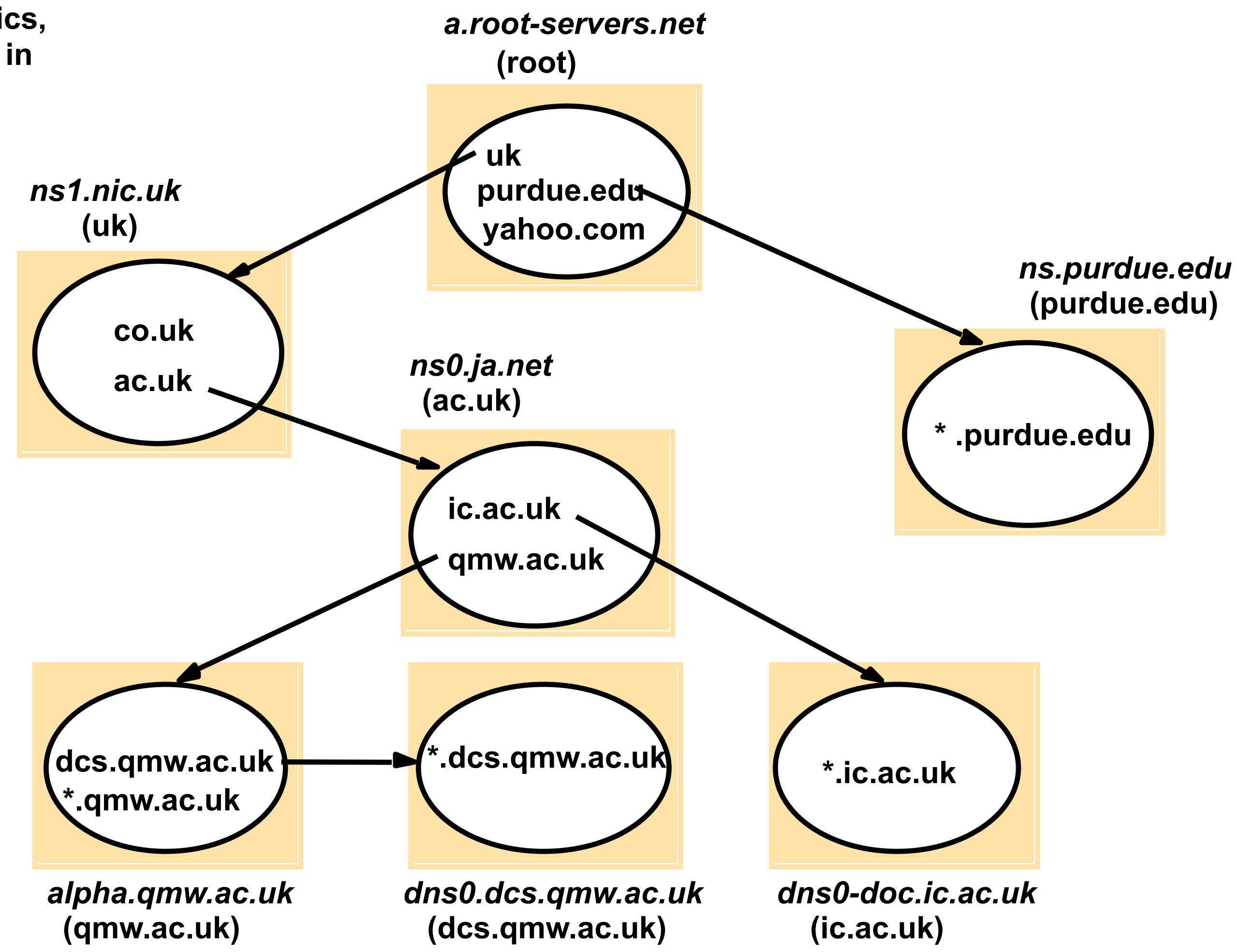
DNS name servers

- Why not centralize DNS?
  - single point of failure
  - traffic volume
  - distant centralized database
  - doesn’t *scale!*

- no server has all name-to-IP address mappings
- local name servers:
  - each ISP, company has *local (default) name server*
  - host DNS query first goes to local name server
- authoritative name server:
  - for a host: stores that host’s IP address, name
  - can perform name/address translation for that host’s name

DNS Name Servers

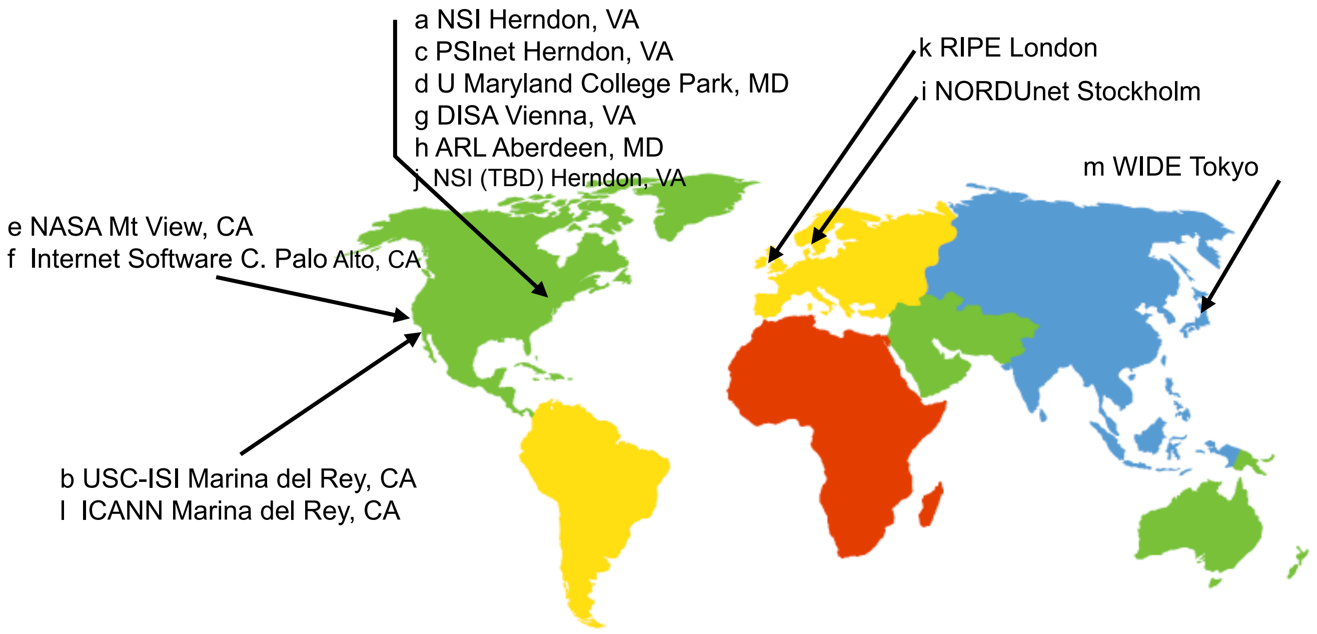
Note: Name server names are in italics, and the corresponding domains are in parentheses.
Arrows denote name server entries



**Note: root server data replicated to about 1000 secondary servers.
Still, each one of them has to process up to 1000 queries per second.**

DNS: Root name servers

- contacted by local name server that can not resolve name
- root name server:
  - contacts authoritative name server if name mapping not known
  - gets mapping
  - returns mapping to local name server



13 root name servers worldwide

DNS Queries

- Host name resolution: Translate symbolic host names into IP addresses

```
% nslookup www.informatik.uni-freiburg.de

Server: avalon.informatik.uni-freiburg.de
Address: 132.230.150.1

Name: falcon.informatik.uni-freiburg.de
Address: 132.230.167.230
Aliases: www.informatik.uni-freiburg.de
```

DNS Database Partitioning

- DNS database partitioned over set of interconnected servers (**partitioning**)

- DNS servers: responsible for a domain
  - locality of requests: most DNS queries are served locally
  - DNS server stores other domain names and the responsible DNS servers

DNS: Zones and Replication

- DNS naming data zones:
- **Authoritative DNS server**: reasonably up to date

- Authoritative DNS servers ("reasonably up to date"), two per zone:

  – **primary server**: reads data directly from master file for that zone, entered by system administrator.

  – **secondary server**: periodically download zone data from primary server typically every few hours or once a day.

DNS Server Caching

- DNS server **caches** query results.
- Will supply results if same query re-occurs.

  - data will be marked as non-authoritative.

  - will do caching within the time to live

  - cache entries timeout (disappear) after some time

- DNS database may contain inconsistent and stale data (order of days…).
- Inconsistency tolerable until stale address data is *used.*
- DNS does not specify how to detect and remedy staleness of address data.

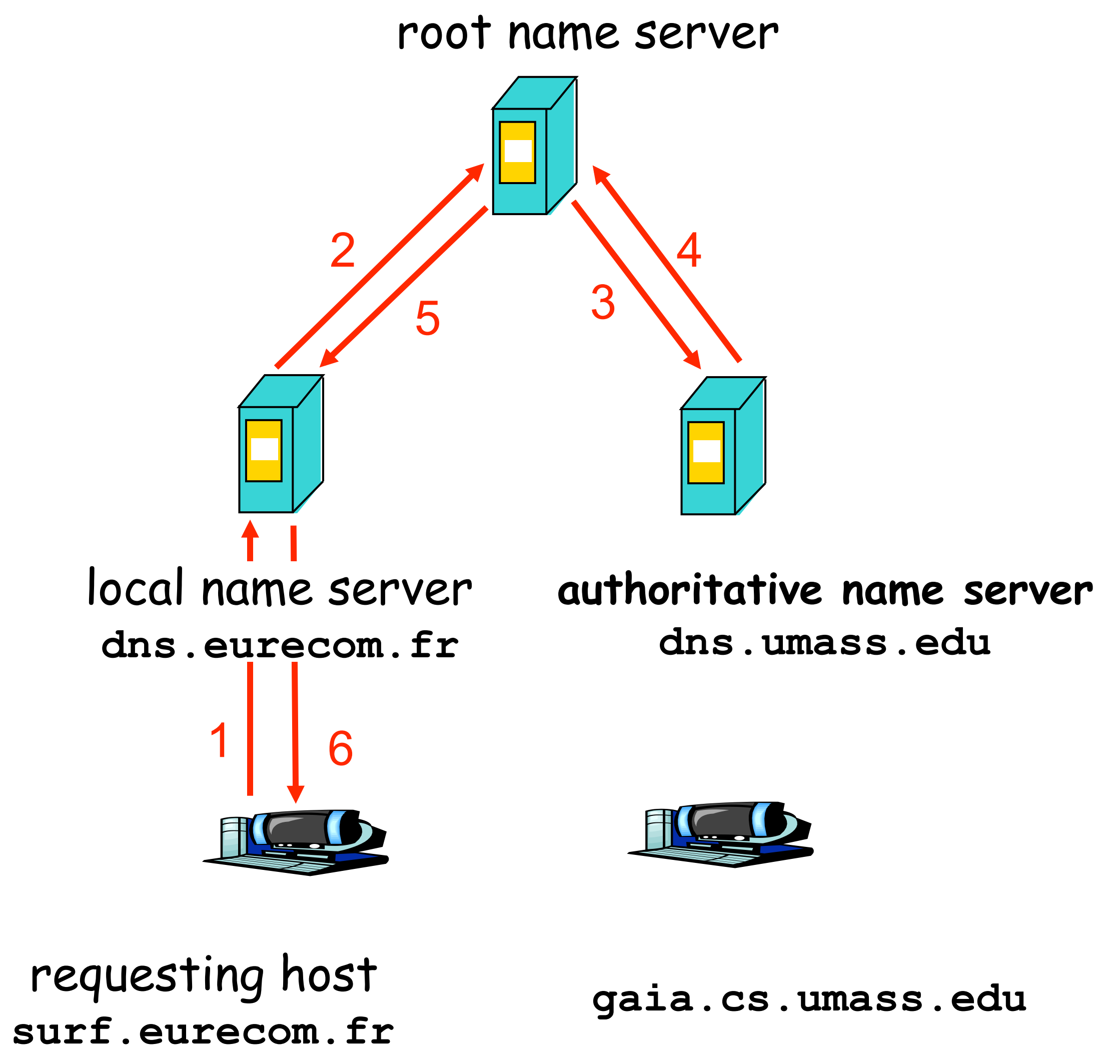
DNS Resolvers

- DNS clients.
- Handles requests to server, usually by UDP-based request/reply protocol.
- May be configured to contact alternate servers if primary choice unavailable.
- Specifies type of navigation to be used by server.
- **Iterative** and **recursive** navigation strategies are all allowed.
- Name servers: are not bound to implement recursive navigation.

Simple DNS example

host **surf.eurecom.fr** wants the IP address of **gaia.cs.umass.edu**

1. contacts its local DNS server, **dns.eurecom.fr**
2. **dns.eurecom.fr** contacts root name server, if necessary.
3. root name server contacts authoritative name server, **dns.umass.edu**, if necessary



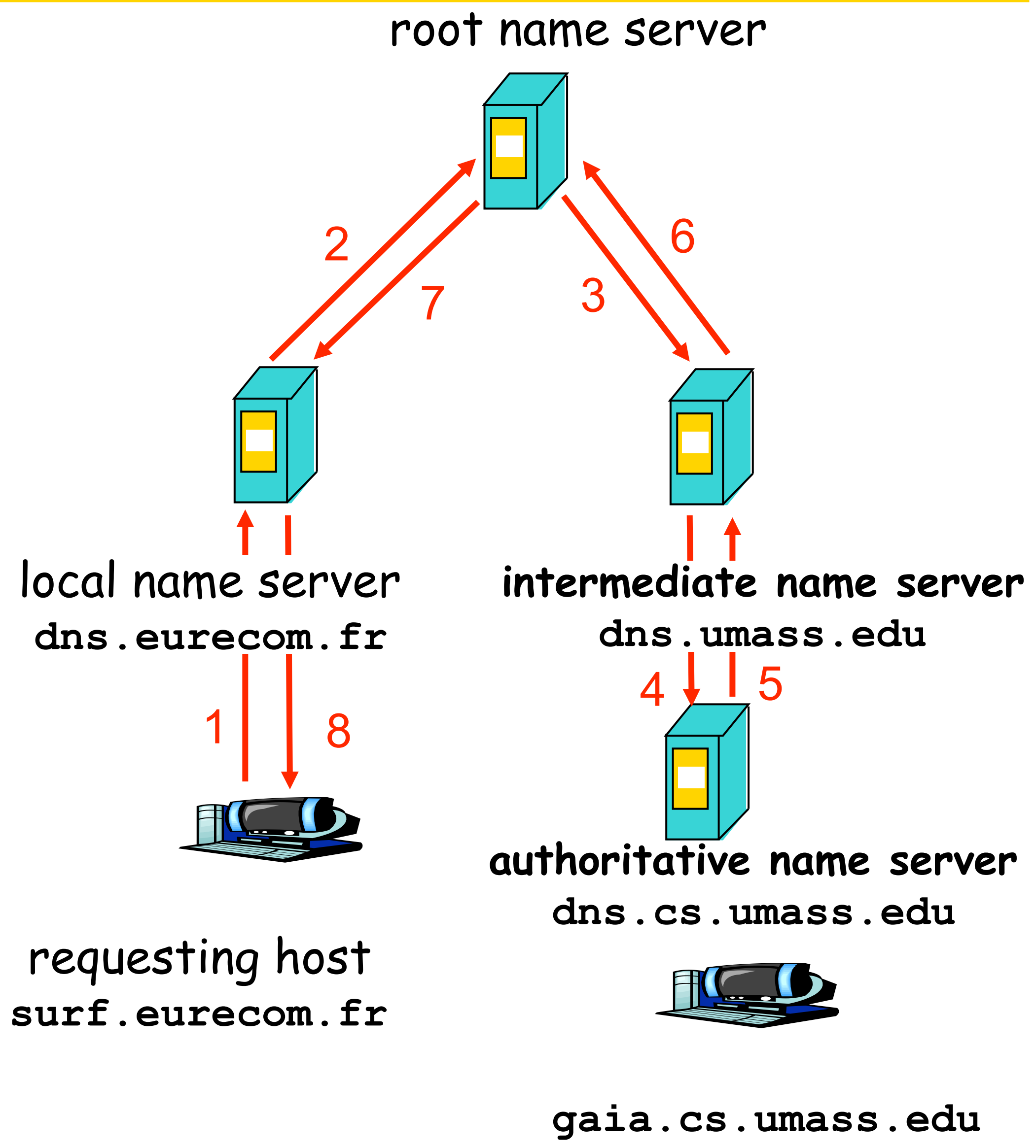
DNS example

-local name server

-root name server

-intermediate name server

-authoritative name server



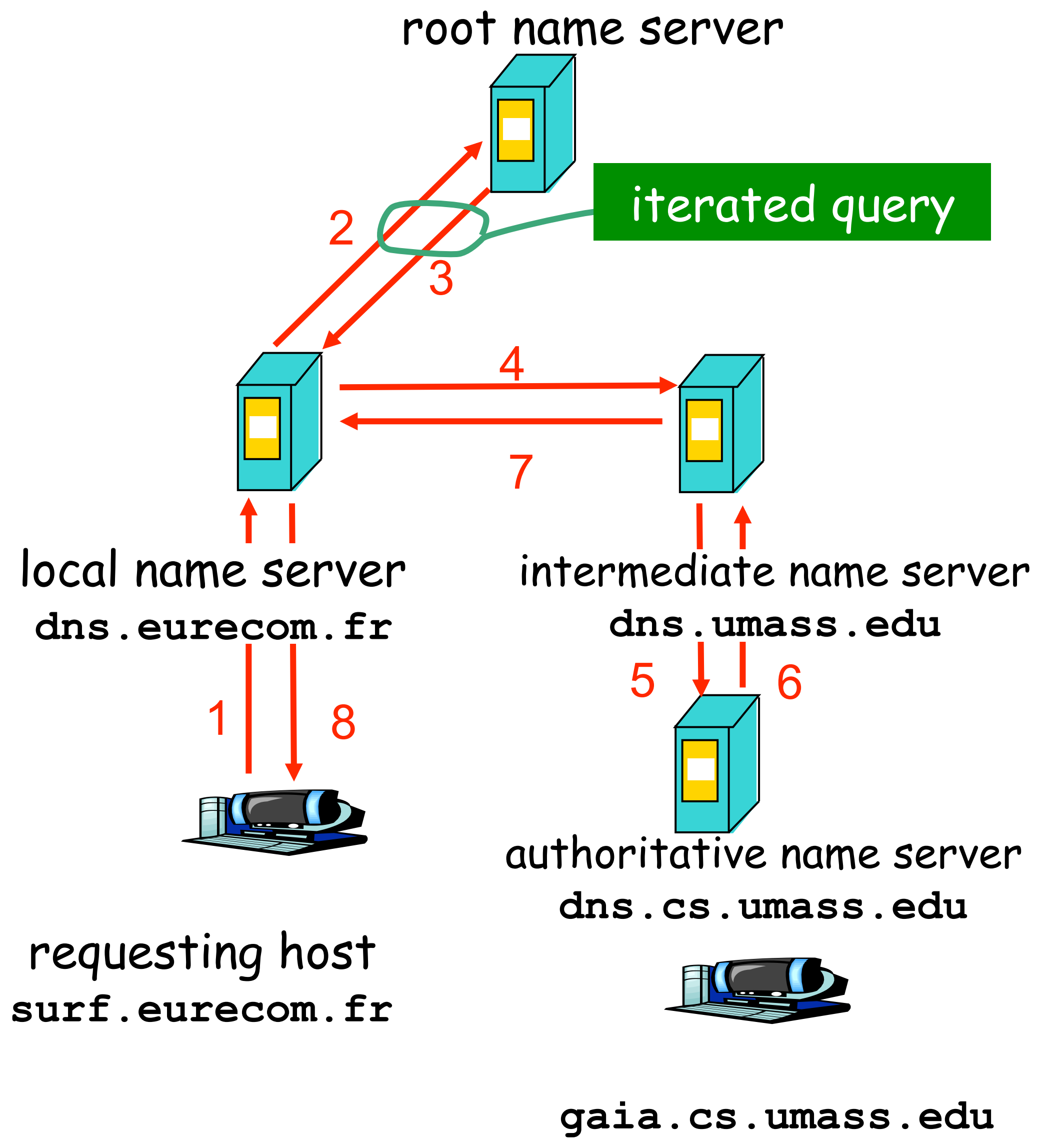
- Root name server:

- may not know authoritative name server

- may know *intermediate name server:* some server to contact to find authoritative name server

DNS: iterated queries

- Recursive query:
  - puts burden of name resolution on contacted name server
  - heavy load?
- Iterated query:
  - contacted server replies with name of server to contact
  - "I don't know this name, but ask this server"



Directory Services

Directory Services

- <name,attributes>:

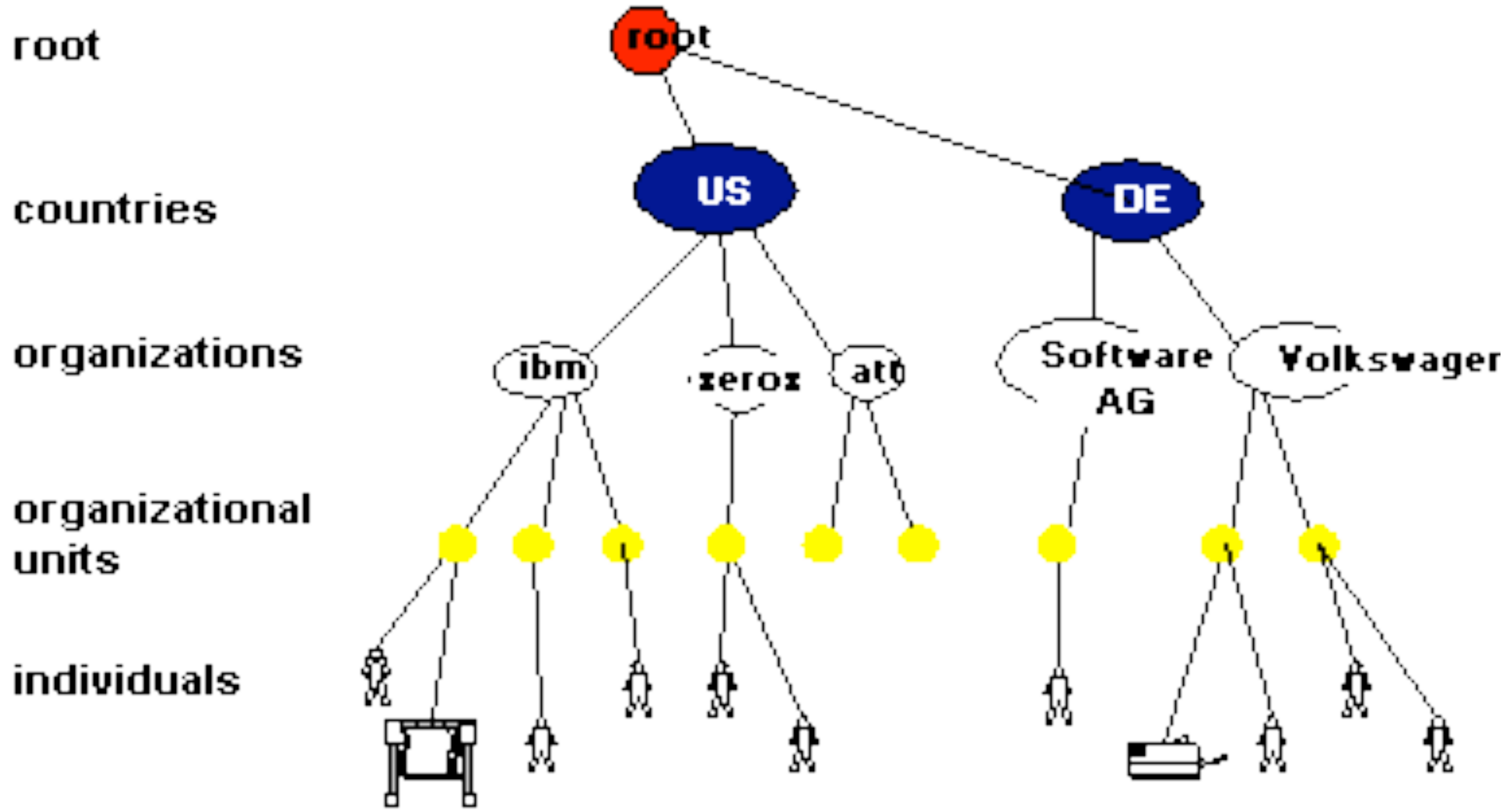
  - search by attribute.
- Directory Service: stores collection of bindings between names and attributes.
- Also called Yellow-Page Services of Attribute-based Name Services:
- Examples:
  - Microsoft Active Directory
  - X.500
  - LDAP

LDAP Fundamentals

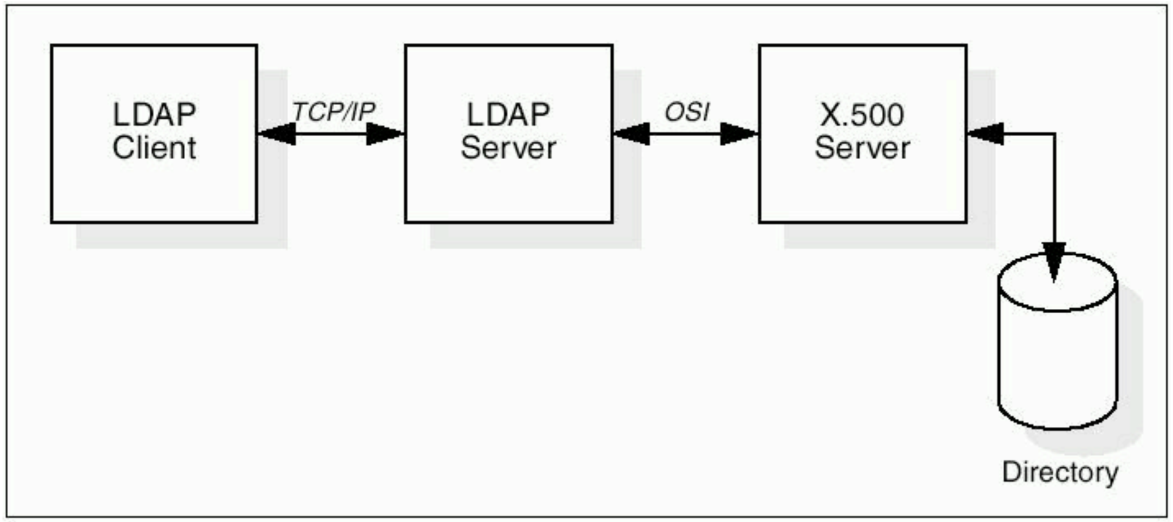
- Lightweight Directory Access Protocol
- Can be directly used in Java: JNDI API
- A way to find people and resources on a network
- Add to, modify, delete from and search directories

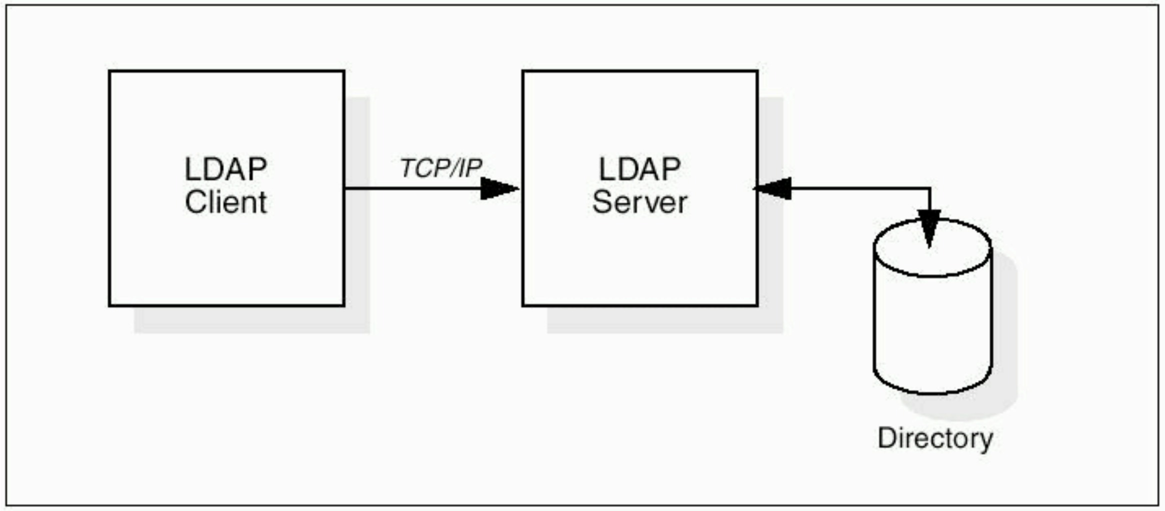
- Each entry has a Distinguished Name (DN) that has a series of attributes (key attributes).
  - e.g. c=Portugal, o=UnivCoimbra, ou=DEI, cn=Luis Silva
- An attribute describes a characteristic of an object.
- Objects reside in a Directory

LDAP Hierarchy (X.500 Directory)

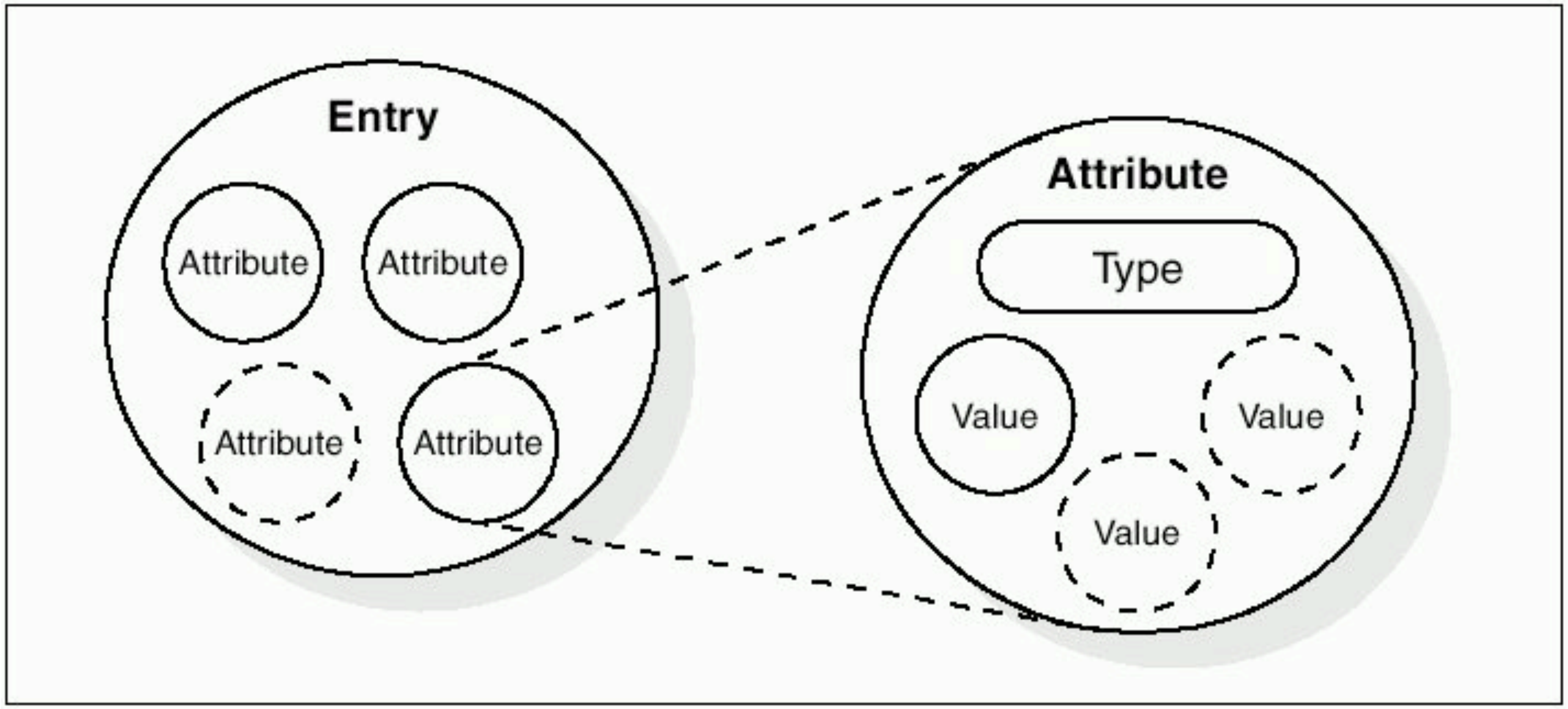


LDAP / X.500


LDAP Client
TCP/IP
LDAP Server
OSI
X.500 Server
Directory


LDAP Client
TCP/IP
LDAP Server
Directory

LDAP Entries



LDAP Attributes

- Each attribute has a type/syntax and a value
- Can define how values behave during searches/directory operations
- Each ‘entry’ describes an object (Class)
  - Person, Server, Printer etc.

Attribute Type	String
CommonName	CN
LocalityName	L
StateorProvinceName	ST
OrganizationName	O
OrganizationalUnitName	OU
CountryName	C
StreetAddress	STREET
domainComponent	DC
Userid	UID

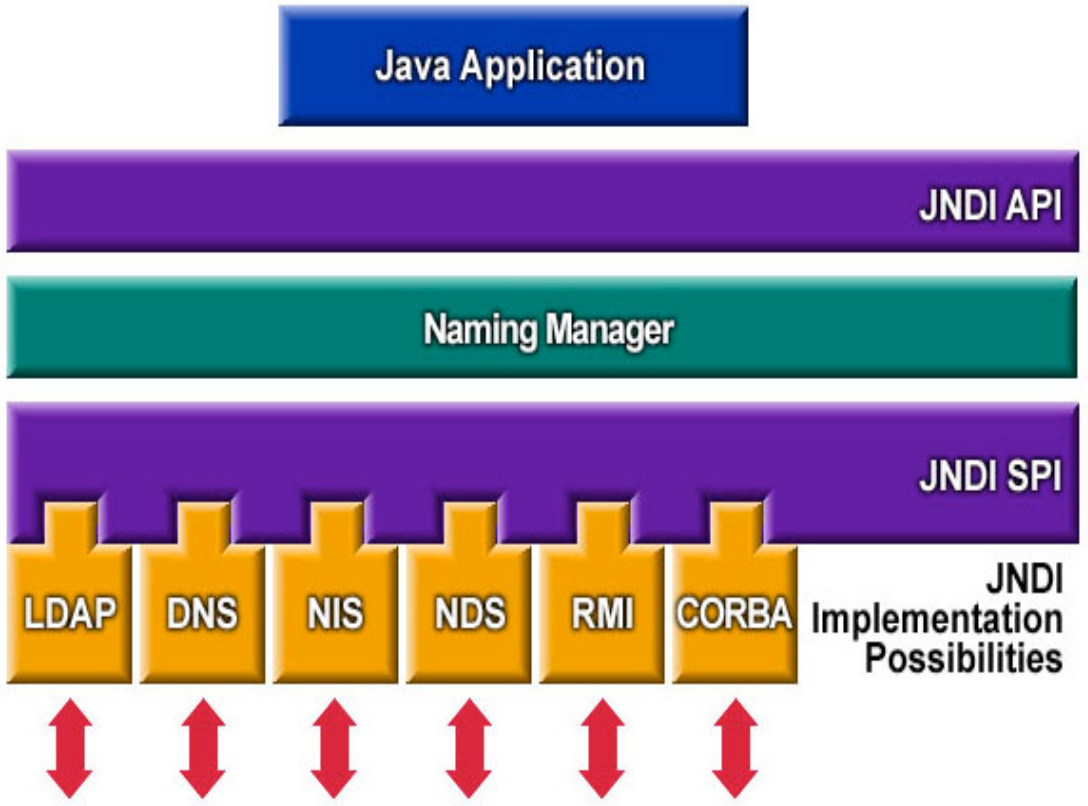
LDAP Functions

- **Authentication**
  - BIND/UNBIND
  - ABANDON
- **Query**
  - Search
  - Compare entry
- **Update**
  - Add an entry
  - Delete an entry (Only Leaf nodes, no aliases)
  - Modify an entry, Modify DN/RDN

LDAP: Client/Server Session

- **Client establishes TCP-IP session with server (BIND)**
  - Hostname/IP and port number
  - Security
    - User-id/password based authentication
    - Anonymous connection - default access rights
    - Encryption/Kerberos also supported
- **Client performs operations**
  - Read/Update/Search
  - SELECT X,Y,Z FROM PART_OF_DIRECTORY
- **Client ends the session (UNBIND)**
- **Client can ABANDON the session**

JNDI API

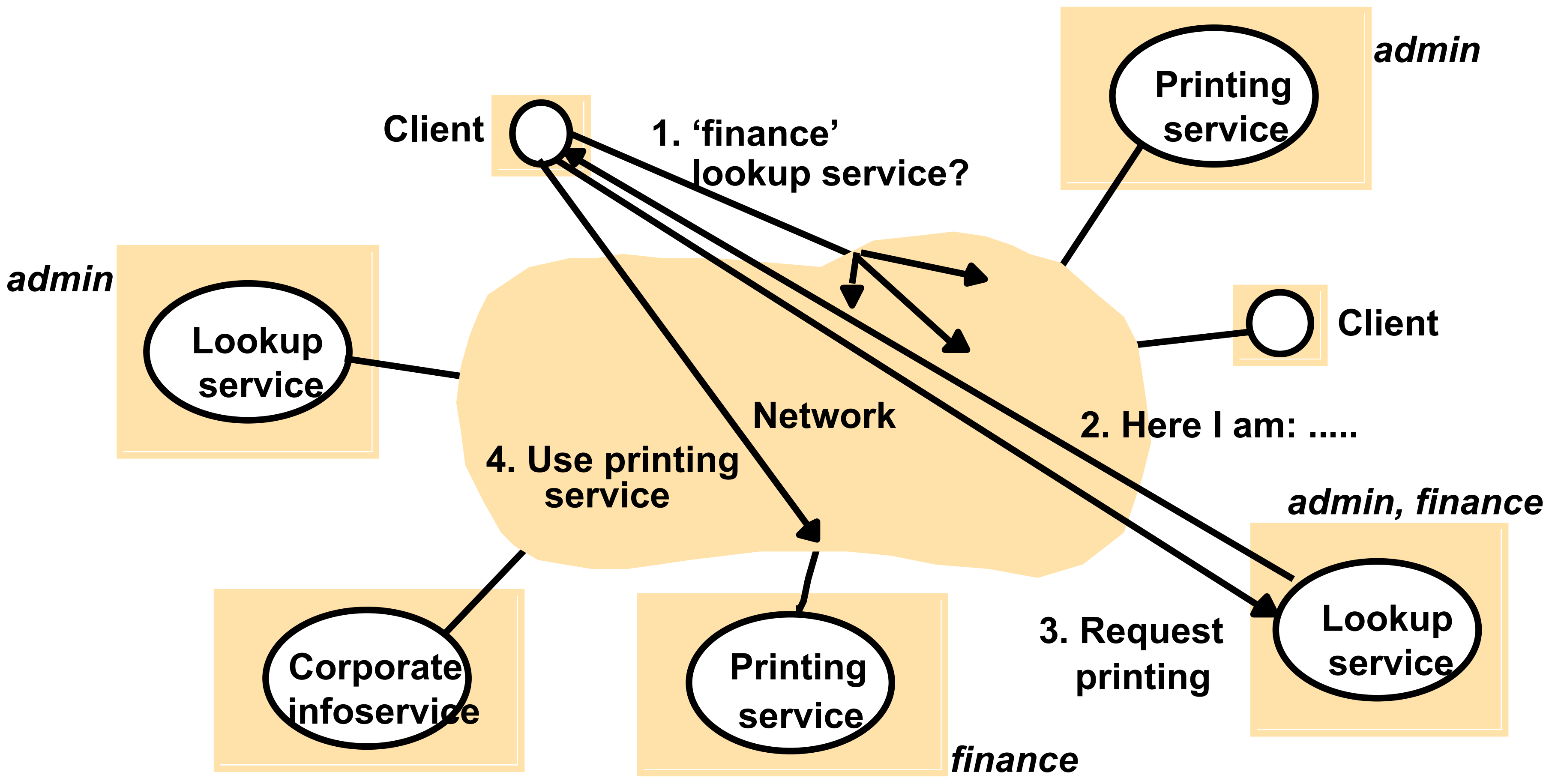

Java Application
JNDI API
Naming Manager
JNDI SPI
LDAP
DNS
NIS
NDS
RMI
CORBA
JNDI Implementation Possibilities

Discovery Services

Discovery Services

- Is a Directory Service that registers the services provided in a spontaneous networking environment.
- Operations:
  - Register
  - Lookup
- **JINI Discovery Service**:
  - Lookup operation uses multicast to a well-known IP-multicast address
  - Register service operation will be provided with a lease that guarantees the registration entry for a limited period of time. If the service does not renew that lease before the expiry time that registration entry will be garbage-collected.

Service Discovery in Jini



UDDI: Directory of Web-Services

UDDI

- Web services is for business-to-business integration.
- For example, company X might expose an invoicing Web service for the use of the suppliers to send electronic invoices.
- Similarly, a vendor V might expose a Web service for placing orders electronically.
- If company X wanted to purchase computer equipment electronically, it would need to search for all vendors who sell computer equipment electronically.
- To do this, company X needs a Yellow-Pages-type directory of all businesses that expose Web services.
- This directory is called **UDDI** (Universal Description, Discovery, and Integration).

Using the UDDI-API

- To invoke these APIs, you send SOAP messages with the appropriate body content.
- For example, to search for a company called XYZ Industries you would send the following XML in the body of a SOAP message:

```
<find_business generic='1.0' xmlns='urn:uddi-org:api'>
      <name>XYZ Industries</name>
</find_business>
```

- The SOAP response that comes back from UDDI contains all businesses that match your search criteria and the registered services for each business.