Naming

CS432: Distributed Systems
Spring 2017

Reading

- Chapter 13 (13.1, 13.2, 13.3, 13.5) [Coulouris '11]
- Chapter 5 (5.1, 5.3, 5.4) [Tanenbaum '06]

Outline

- **Names, Identifiers, and Addresses**
- Name Services
- Case Study: Domain Name System (DNS)
- Directory Services
- Case Study: X.500 Directory Service

Objective

- Various compute resources: computers, services, remote objects, files,…
- Objective: refer to these distributed computer resources ==> a namespace for all of them
  - URL to access web pages, domain names
  - A process on another computer
  - Users communicate in a distributed system (e.g. via email)
  - File names (e.g. /etc/passwd)

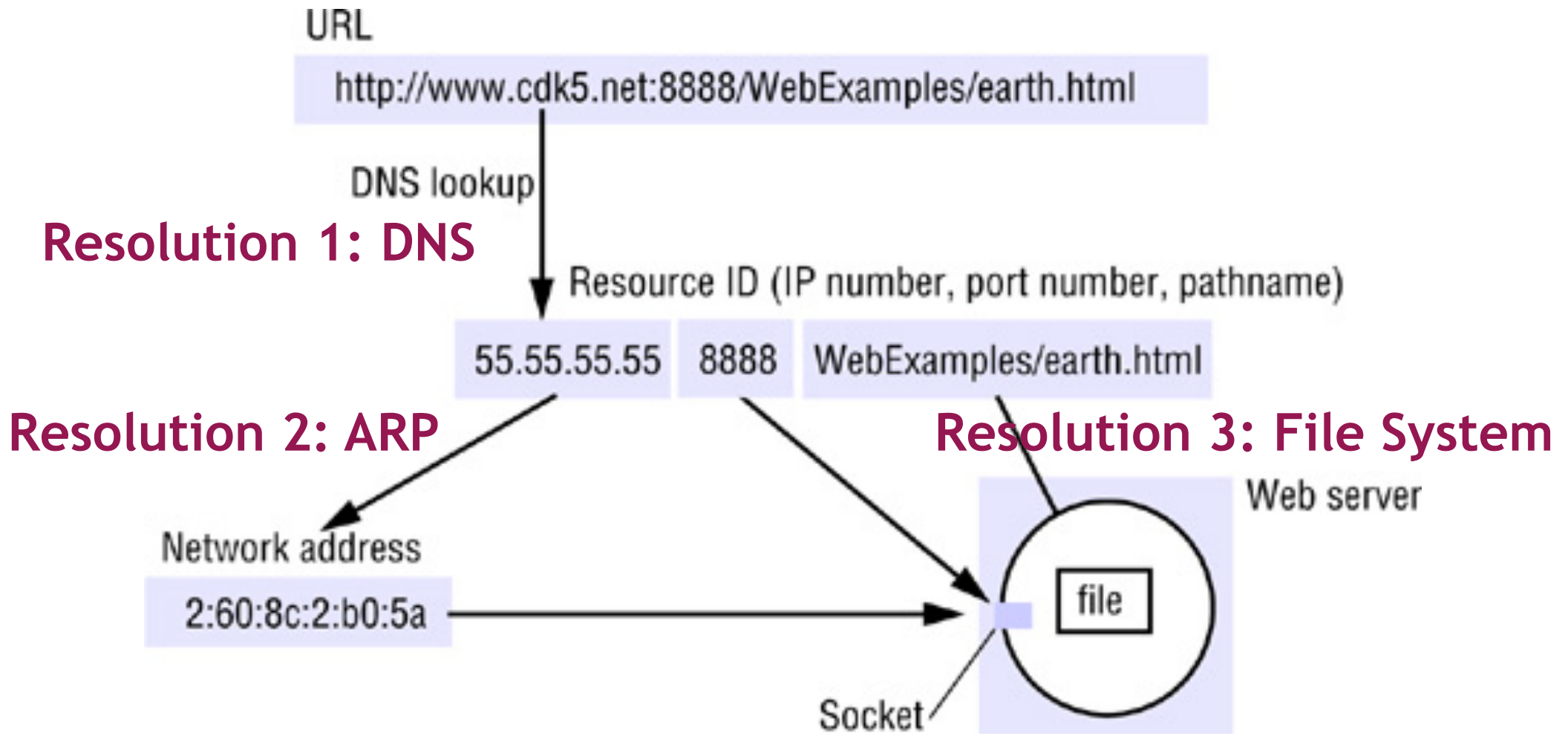
Names, Identifiers, Addresses

- A name in a distributed system is a string of bits or characters that is used to refer to an entity
- Examples of entities: hosts, printers, files, processes, users, mailboxes, …
- Address is a special kind of name; a value that identifies the location of the object rather than the object itself (**not unique**)
  - Same name could be mapped to multiple addresses
- Identifiers: uniquely identify an entity
  - Sometimes used to refer to names that are interpreted only by programs (e.g. NFS file handles)

Resolving Names

- Translating names and identifiers into object/ entity/resource
- Naming system binds name and object (its attributes, mostly address)
- Example of attributes of entities:
  - The DNS maps domain names to the attributes of a host computer: its IP address
  - The X.500 directory service maps a person's name onto attributes including their email address and telephone number

Composed Naming Domains: Access a Resource from a URL



URI, URL, and URN

- Uniform Resource Identifiers (URI): used to identify resources on the web
  - Uniform means that their syntax incorporates indefinitely many individual types of resource identifiers
- Uniform Resource Locators (URL): URIs contain information that can be used to locate and access a resource (e.g.: http://www.cdk5.net/, mailto:fred@flintstone.org)
- Uniform Resource Names (URN): URIs that are used as pure resource names rather than locators (e.g.: mid:0E4FC272-11D9-B115-000A95B55BC8@hpl.hp.com)

Name and Directory Services

- Name Service: stores information about a collection of textual names, in the form of bindings between the names and the attributes of the entities they denote
  - <name, attributes> (e.g. of attribute: IP address)
  - Resolving a name means to return attributes given a name
- Directory Service: provides data about objects that satisfy a given description
  - Store collections of <name, attributes> pairs
  - Returns the sets of attributes of object given specified attributes
  - Request ‘TelephoneNumber = 020 555 9980’ might return {‘Name = John Smith’, ‘TelephoneNumber = 020 555 9980’, ‘emailAddress = john@dcs.gormenghast.ac.uk’, ...}

Outline

- **Names, Identifiers, and Addresses**
- Name Services
- Case Study: Domain Name System (DNS)
- Directory Services
- Case Study: X.500 Directory Service

Name Services

- Characteristics: namespace + name resolution
- Flat naming (e.g. identifiers): random bits with no indication to location
  - Multicast / broadcast
  - Home-based approaches (e.g: Mobile IP)
  - Distributed hash tables (e.g. Chord): resolve key K to successor of k
  - Hierarchical approaches (e.g. Global location system)
- Structured naming (e.g. file names, internet host names): human readable names

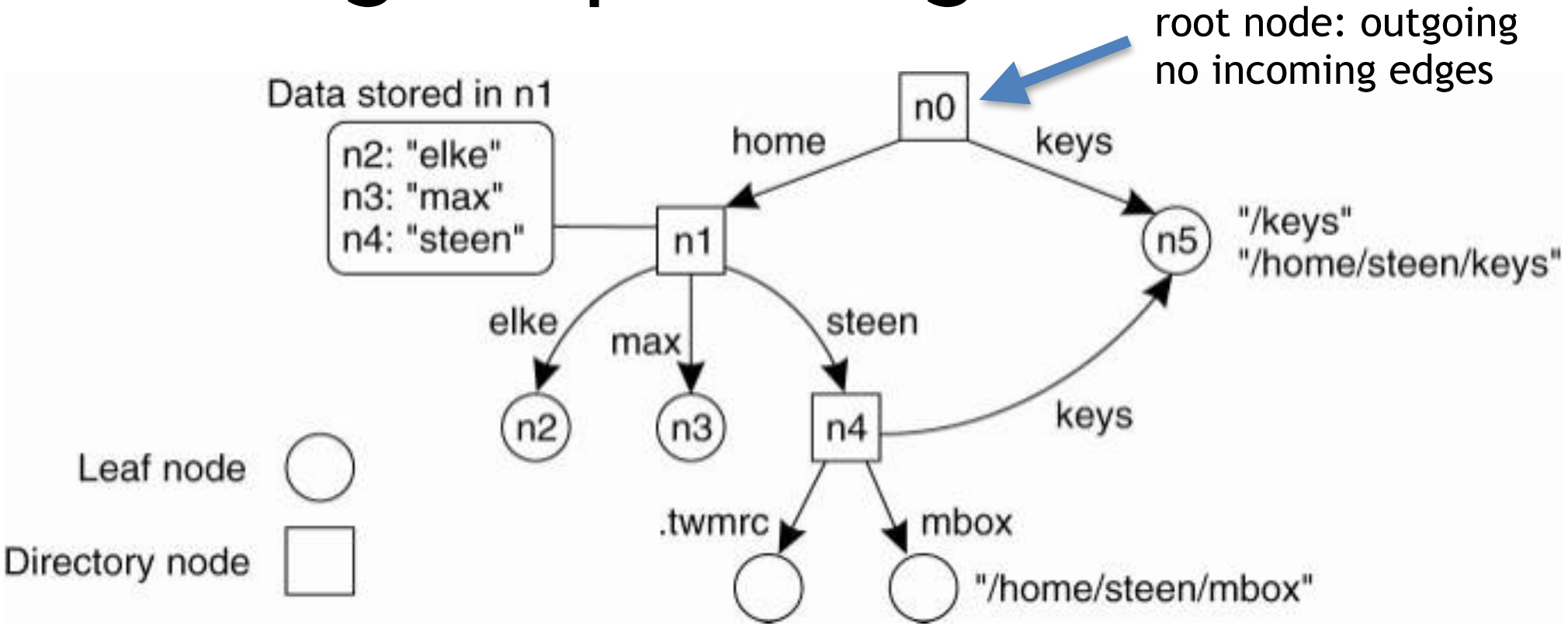
Name Services Requirements

- Handle an arbitrary number of names and to serve an arbitrary number of administrative organizations (scalability)
- Long lifetime (adaptability of changes)
- High availability: many distributed systems depend on it
- Fault isolation: local failures should not cause the entire service to fail
- Performance: fast lookup operations
- Consistency: tolerate short inconsistencies — eventually converge to consistent state

Namespaces

- Namespace: the collection of all valid names recognized by a particular service
  - syntactic definition to recognize invalid names
  - bounded name means that it corresponds to an object
  - a valid name could be unbound
- Advantages of hierarchical namespaces:
  - can grow indefinitely
  - different contexts can be managed by different people or organizations
- Aliases: convenient names are substituted for complicated ones (e.g. URL shorteners)

Naming Graph - Single Root Node



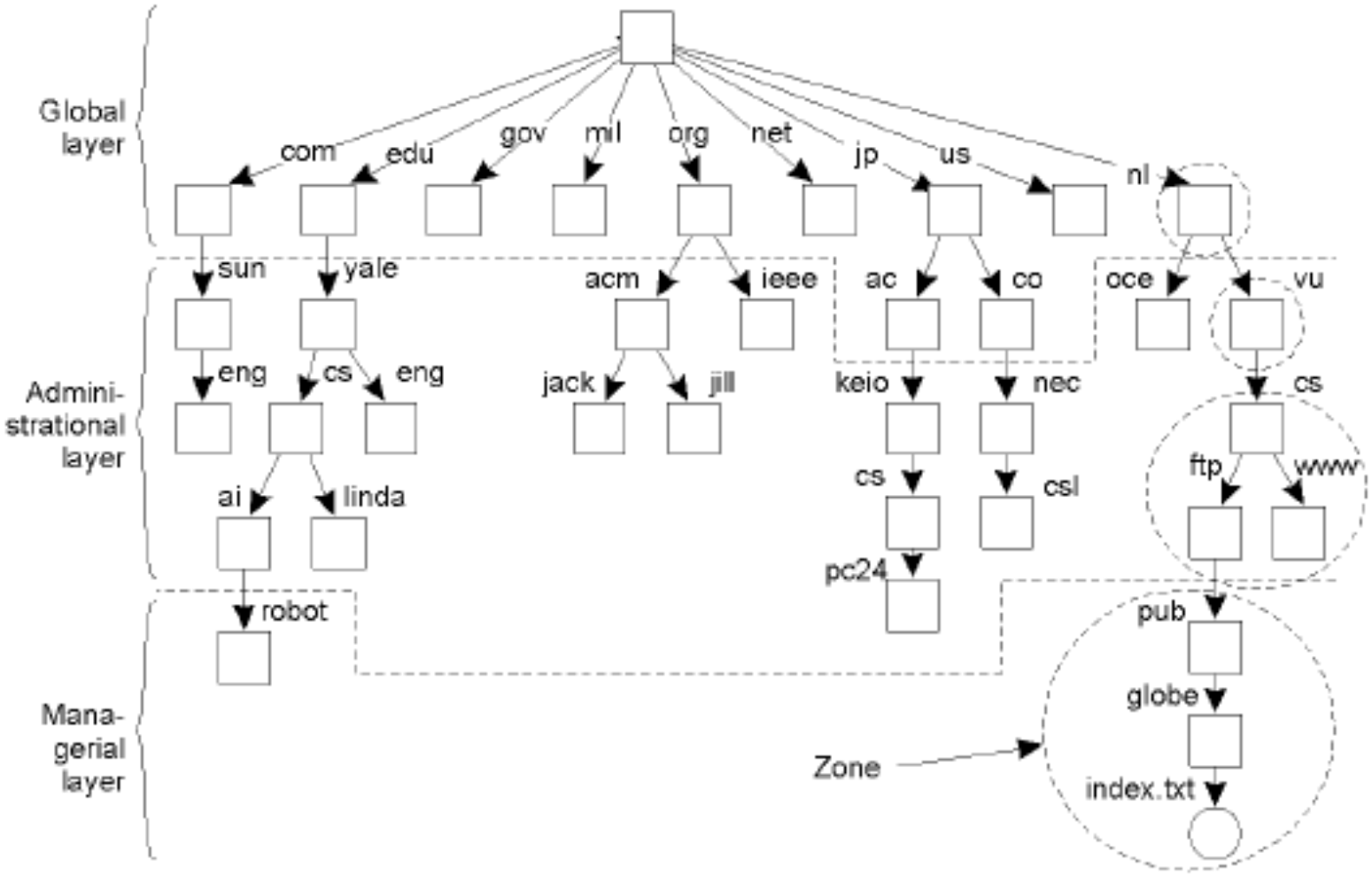
- Namespace represented as a labeled directed graph with two types of nodes
  - leaf node: represents a names entity
  - directory node: keeps a directory table for edge mapping

Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Namespace Distribution

- Organized hierarchically — once single root node
- Partition namespace into logical layers:
  - Global layer: highest level nodes (root + directories close to root), stable, represent organizations
  - Administrational layer: directory nodes that together are managed within a single organization, changes more frequently than global layer
  - Managerial layer: consists of nodes that may typically change regularly — usually maintained by end users of the distributed system
- Zones of DNS: non-overlapping parts of namespace — each implemented by separate name server

Namespace Distribution Example



An example partitioning of the DNS name space, including Internet-accessible files, into three layers.

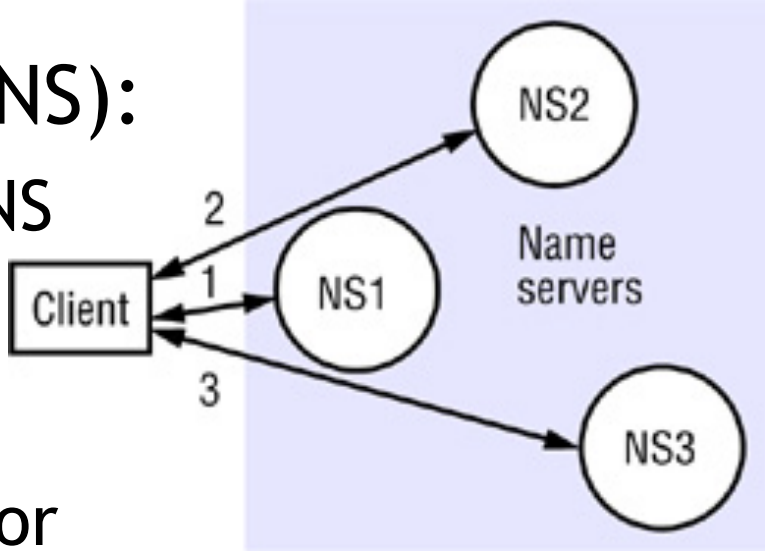
Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Name Resolution

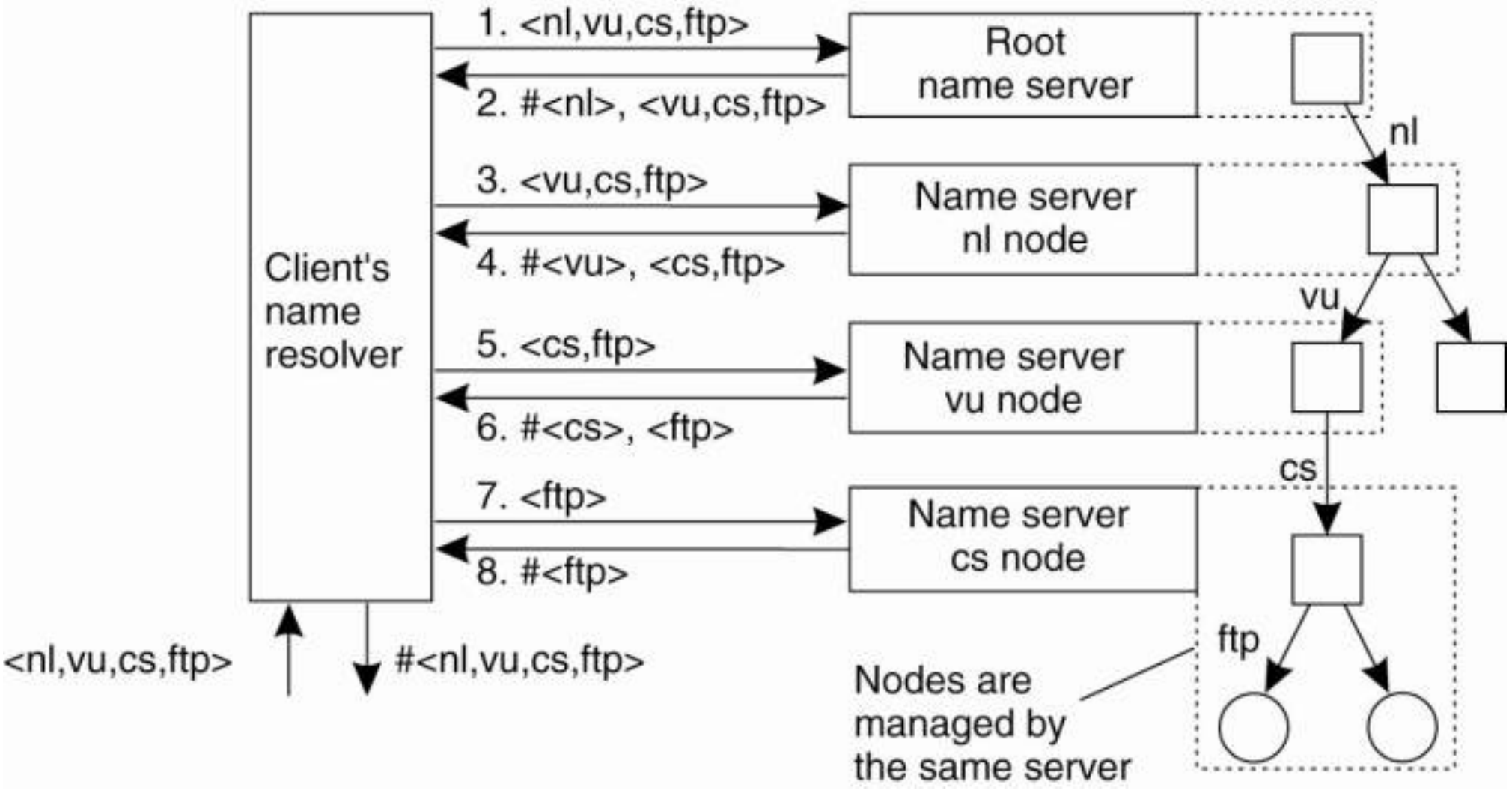
- Name resolution is interactive or recursive process to look up the attributes to which a name refers
- Iteratively - for a given name:
  - maps it onto a set of primitive attributes (done!)
  - maps it onto a further naming context and derived name
- Using alias, first resolve the alias to a domain name
- Cycles: abandon resolution after a threshold number of resolutions (or leave it to admins!)

Navigation

- Design characteristic: a name service does not store its database on a single server
- **Navigation**: the process of locating naming data from more than one name server in order to resolve a name
- Iterative navigation (e.g. in DNS):
  - Client SW sends name to local NS
  - A NS returns result or suggests another server
  - Continue until name is located or is discovered to be unbound



Example: Iterative Navigation

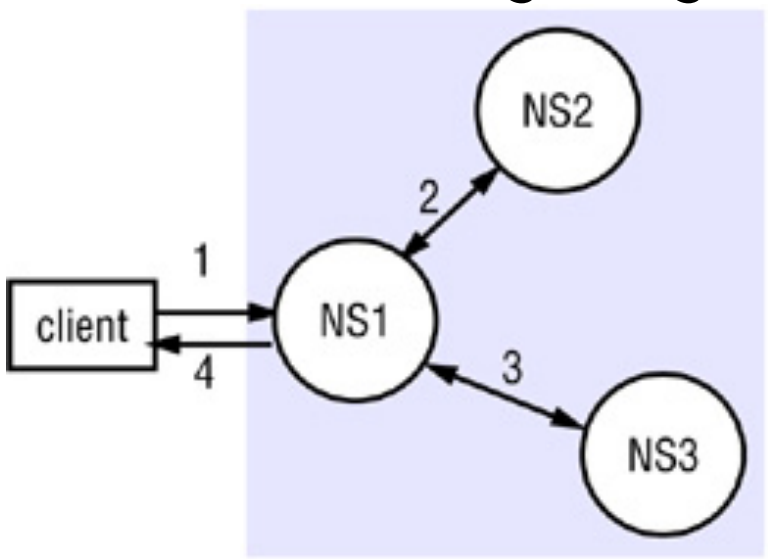


#<nl> indicates the address of server responsible for handling node <nl>

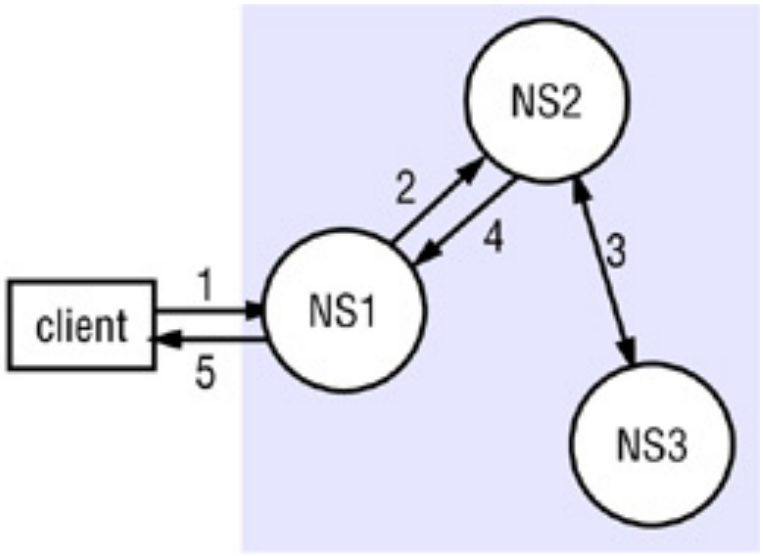
Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Other Navigation Approaches

- Non-recursive server-controlled navigation
  - If server cannot resolve the name, it contacts another server (multicast or iterative)
- Recursive navigation
  - If a server cannot resolve the name it contacts another server server storing a larger prefix of the name

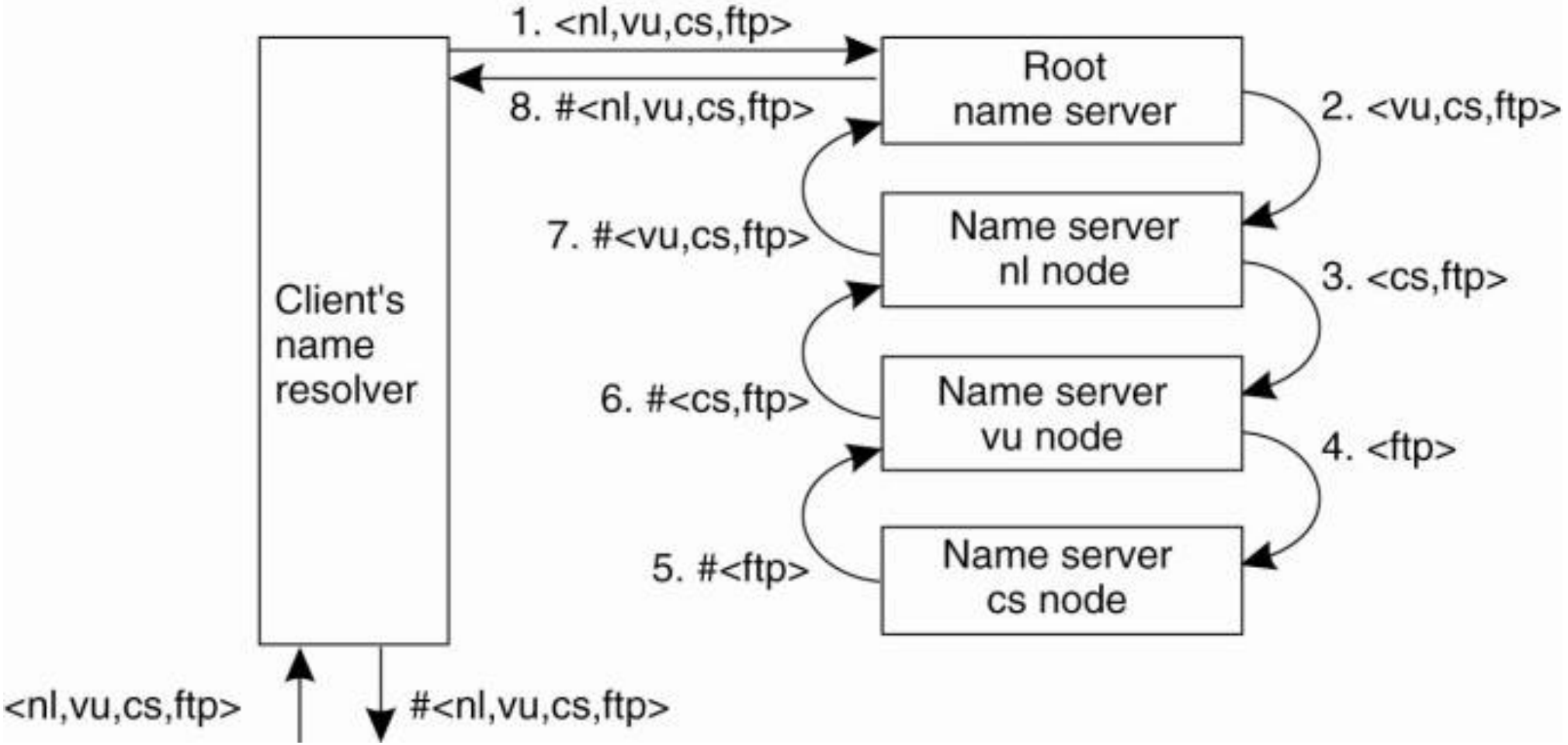


Non-recursive server-controlled



Recursive server-controlled

Example: Recursive Navigation



Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Discussion

- Recursive name resolution puts a higher performance demand on each name server
  - note that a name server is required to handle the complete resolution of a path name — even with help of others servers
  - servers in the global layer of a name space support only iterative name resolution
- Advantages of recursive name resolution
  - Caching results is more effective compared to iterative name resolution
  - Communication cost may be reduced

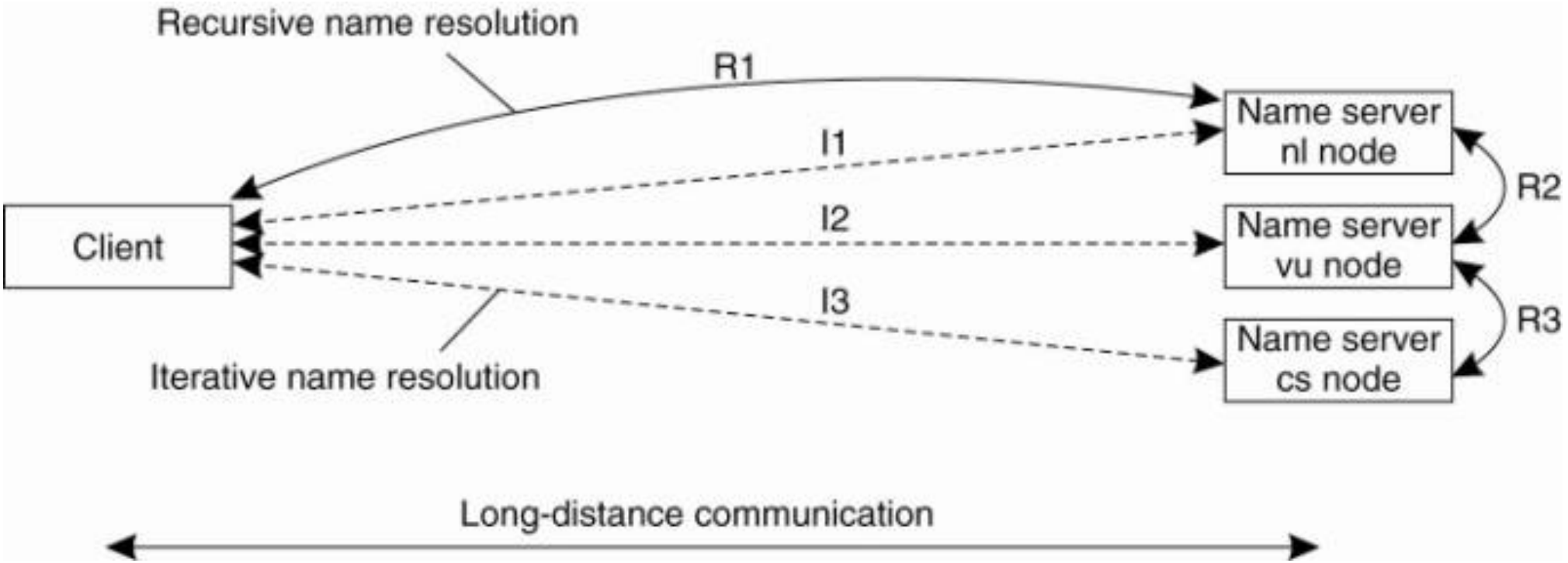
Caching in Recursive Resolution

Server for node	Should resolve	Looks up	Passes to child	Receives and caches	Returns to requester
cs	<ftp>	#<ftp>	–	–	#<ftp>
vu	<cs,ftp>	#<cs>	<ftp>	#<ftp>	#<cs> #<cs, ftp>
ni	<vu,cs,ftp>	#<vu>	<cs,ftp>	#<cs> #<cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>
root	<ni,vu,cs,ftp>	#<nl>	<vu,cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>	#<nl> #<nl,vu> #<nl,vu,cs> #<nl,vu,cs,ftp>

Recursive name resolution of *<nl, vu, cs, ftp>*. Name servers cache intermediate results for subsequent lookups

Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Communication Cost of Navigation



The comparison between recursive and iterative name resolution with respect to communication costs

Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Domain Name System (DNS)

- DNS is a name service design whose main naming database is used across the Internet
- Usage: looking up IP addresses of hosts and mail servers
  - Stored object are primarily computers and one of their attributes are IP addresses
- Hierarchically organized as a rooted tree
- Each node has one incoming edge (except root) => label of edge is also name of node
- Domain: a subtree
- Domain name: a path name to its root

Why DNS is Distributed

- Old centralized approach:
  - A central master server holds a file with all host names and addresses
  - Download file and then resolve host names locally
- DNS vs old centralized approach
  - Centralized DNS does not scale (single point of failure + high traffic)
  - Centralized administration vs multiple entitles administering their part of naming service
  - General name service vs only for internet hosts

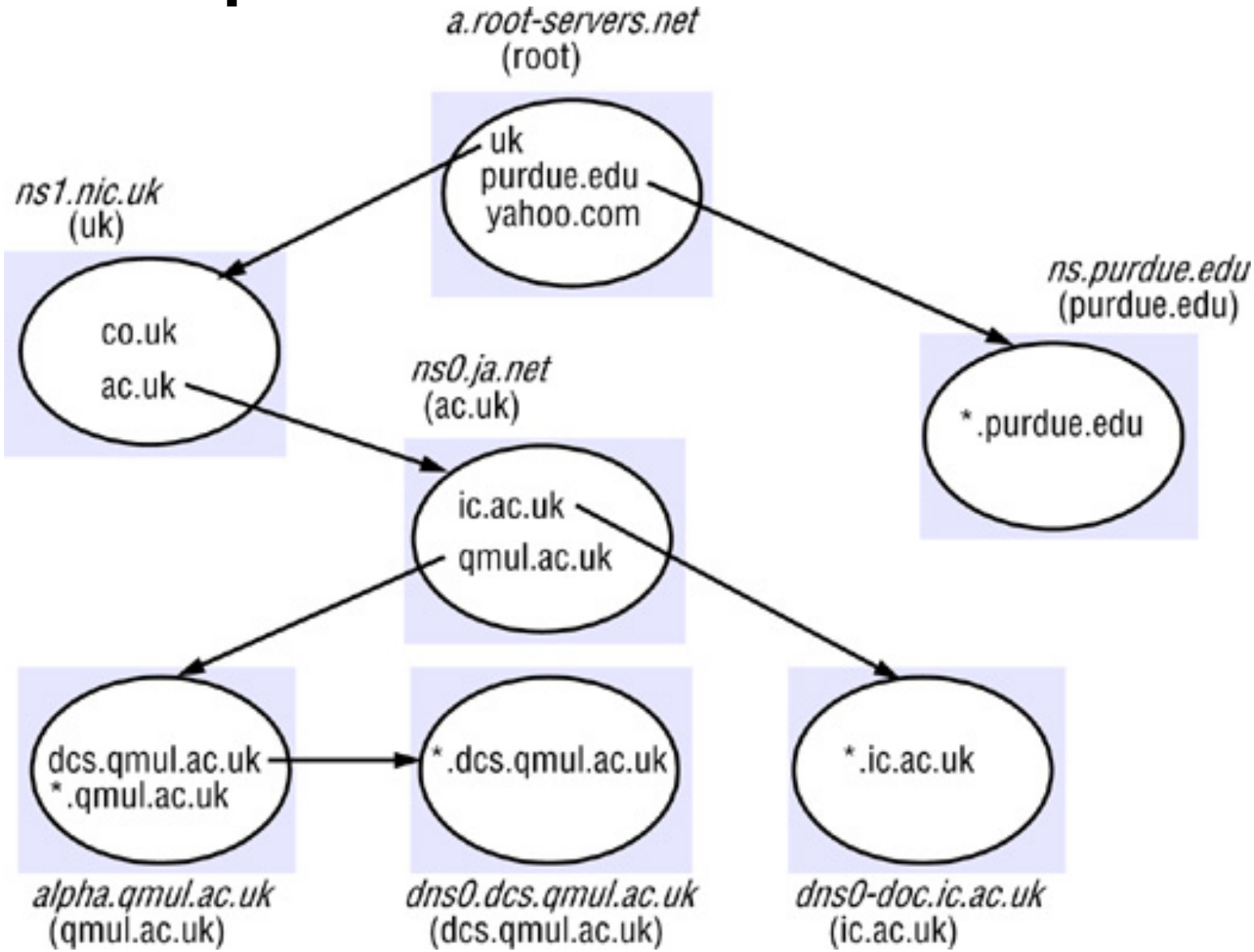
DNS Queries

- Host name resolution
  - Resolve host name to IP address
  - Use one of the protocols (HTTP, FTP) to communicate with the server at the given IP address
- Mail host location
  - Resolve domain name to IP address of mail host
  - Returned value can include multiple hosts, that the mail client can try contact (note: one mail service per domain)
- Reverse resolution
  - Query a name given IP address. Name server must be the right domain
- Host information: e.g. machine architecture and operating system

DNS Name Servers

- Scaling: partitioning + replication + caching
- DNS database is distributed across a logical network of servers
  - Each server holds part of the naming database (data for the local domain)
  - Each server can answer queries about hosts in that domain and names and addresses of other name servers
- Data stored by a zone:
  - names in this domain less data administered by lower levels
  - At least two servers that can provide authoritative data for the zone
  - Names of name servers holding data for subdomains
  - Zone management parameters (caching + replication)

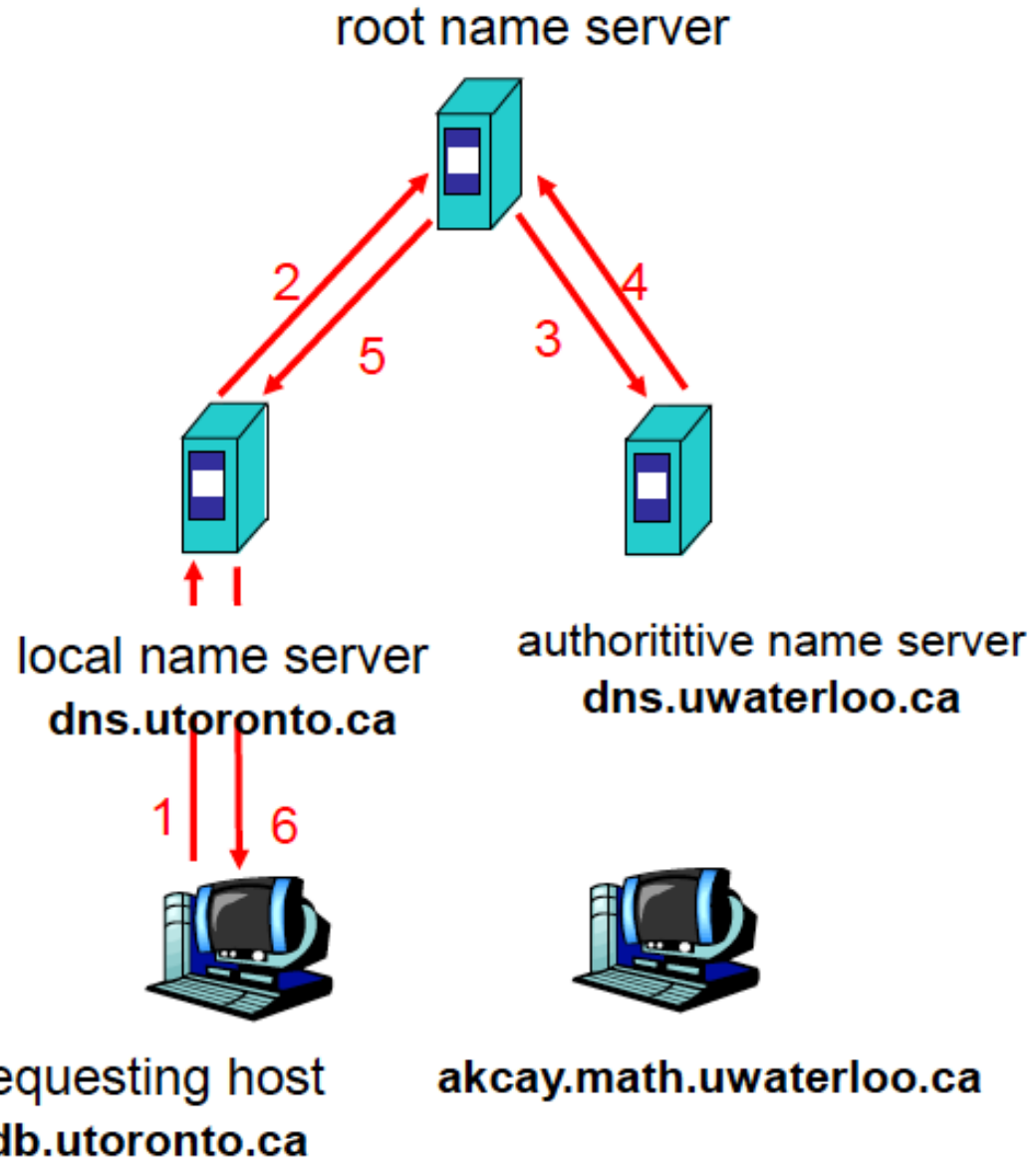
Example: DNS Name Servers



DNS Example 1

Host **db.utoronto.ca** wants IP address of **akcay.math.uwaterloo.ca**

1. Contacts its local DNS server, **dns.utoronto.ca**
2. **dns.utoronto.ca** contacts root name server, if necessary
3. root name server contacts authoritative name server, **dns.uwaterloo.ca,** if necessary

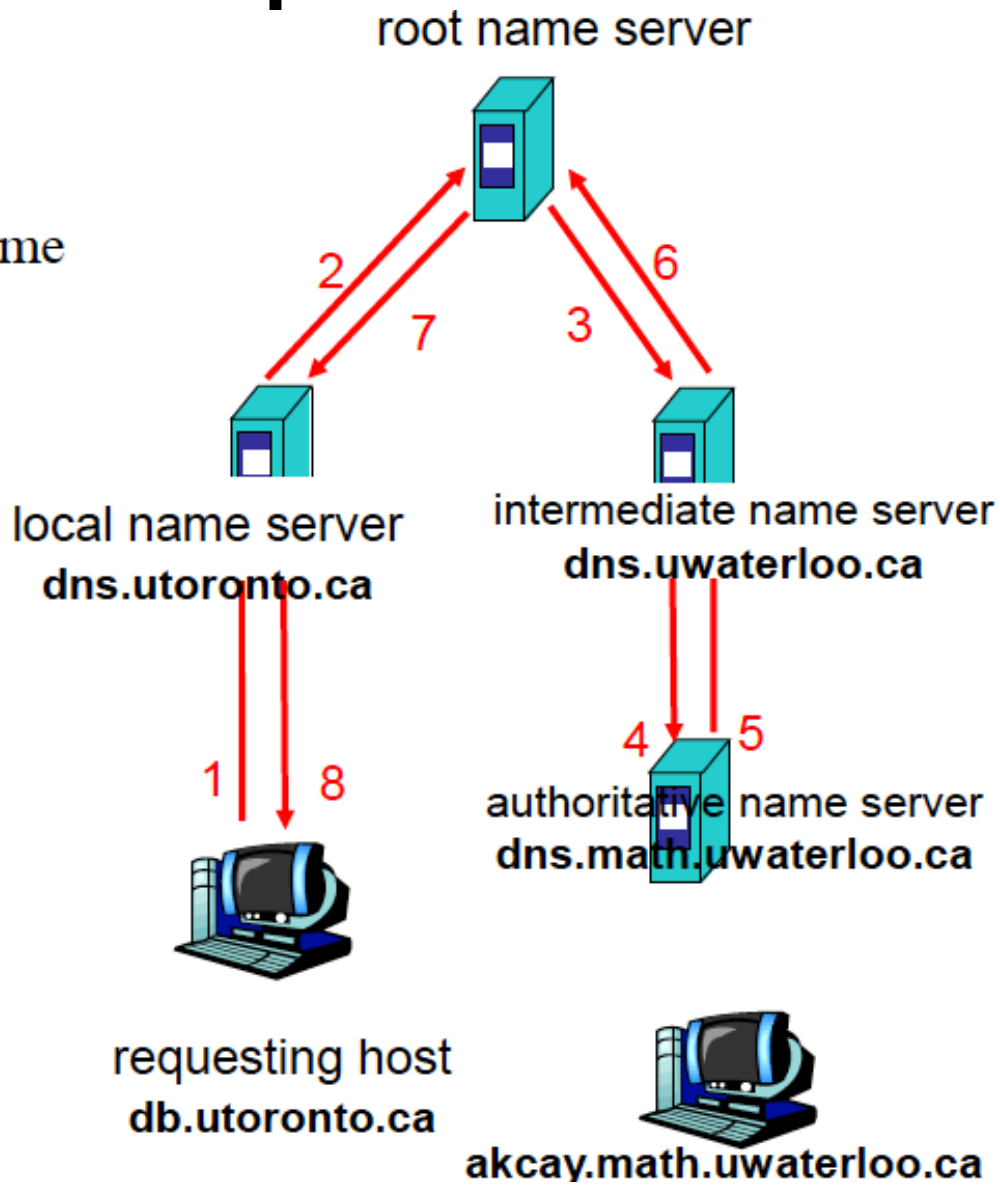


Credit: UW Lecture Notes

DNS Example 2

Root name server:

- may not know authoritative name server
- may know *intermediate name server:* who to contact to find authoritative name server



Credit: UW Lecture Notes

DNS Record Types

Record type	Meaning	Main contents
A	A computer address (IPv4)	IPv4 number
AAAA	A computer address (IPv6)	IPv6 number
NS	An authoritative name server	Domain name for server
CNAME	The canonical name for an alias	Domain name for alias
SOA	Marks the start of data for a zone	Parameters governing the zone
PTR	Domain name pointer (reverse lookups)	Domain name
HINFO	Host information	Machine architecture and operating system
MX	Mail exchange	List of <preference, host= pairs
TXT	Text string	Arbitrary text

DNS Resource Records

- Format of resource record:
  - domain name, time to live, class, type, value
- Data for a zone:
  - SOA record (parameters for this zone)
  - NS records: name servers for the domain
  - MX records: mail servers for the domain
  - A records: IP addresses for domain names / hosts

DNS DB for Zone cs.vu.nl

Name	Record type	Record value
cs.vu.nl	SOA	star (1999121502,7200,3600,2419200,86400)
cs.vu.nl	NS	star.cs.vu.nl
cs.vu.nl	NS	top.cs.vu.nl
cs.vu.nl	NS	solo.cs.vu.nl
cs.vu.nl	TXT	"Vrije Universiteit - Math. & Comp. Sc."
cs.vu.nl	MX	1 zephyr.cs.vu.nl
cs.vu.nl	MX	2 tornado.cs.vu.nl
cs.vu.nl	MX	3 star.cs.vu.nl
star.cs.vu.nl	HINFO	Sun Unix
star.cs.vu.nl	MX	1 star.cs.vu.nl
star.cs.vu.nl	MX	10 zephyr.cs.vu.nl
star.cs.vu.nl	A	130.37.24.6
star.cs.vu.nl	A	192.31.231.42
zephyr.cs.vu.nl	HINFO	Sun Unix
zephyr.cs.vu.nl	MX	1 zephyr.cs.vu.nl
zephyr.cs.vu.nl	MX	2 tornado.cs.vu.nl
zephyr.cs.vu.nl	A	192.31.231.66
www.cs.vu.nl	CNAME	soling.cs.vu.nl
ftp.cs.vu.nl	CNAME	soling.cs.vu.nl
soling.cs.vu.nl	HINFO	Sun Unix
soling.cs.vu.nl	MX	1 soling.cs.vu.nl
soling.cs.vu.nl	MX	10 zephyr.cs.vu.nl
soling.cs.vu.nl	A	130.37.24.11
laser.cs.vu.nl	HINFO	PC MS-DOS
laser.cs.vu.nl	A	130.37.30.32
vucs-das.cs.vu.nl	PTR	0.26.37.130.in-addr.arpa
vucs-das.cs.vu.nl	A	130.37.26.0

Tanenbaum and van Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, Inc. 2002

Outline

- Names, Identifiers, and Addresses
- Name Services
- Case Study: Domain Name System (DNS)
- **Directory Services**
- Case Study: X.500 Directory Service

Directory Services

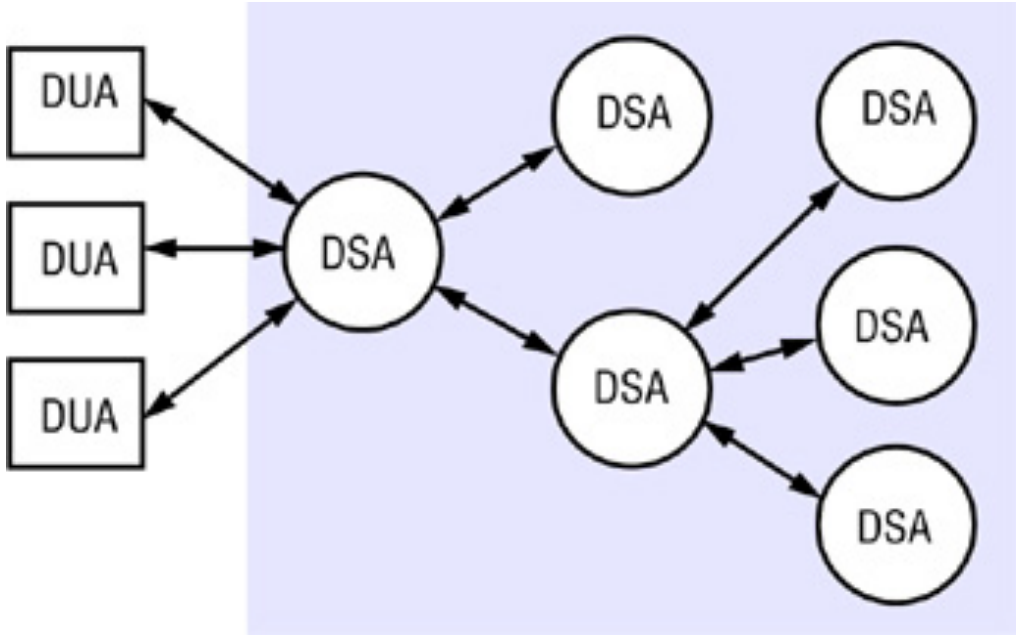
- Attribute-based naming: <attribute , value>
  - Binding between names and attributes
  - Look up entries that match attribute
- Attributes are what we look up
- Examples:
  - Look up a user, whose phone number is xxxxx
  - Look up a machine that has a solaris OS
  - Look up a color printer
- Examples: X.500 and LDAP
- Note: directory service resemble yellow pages

X.500

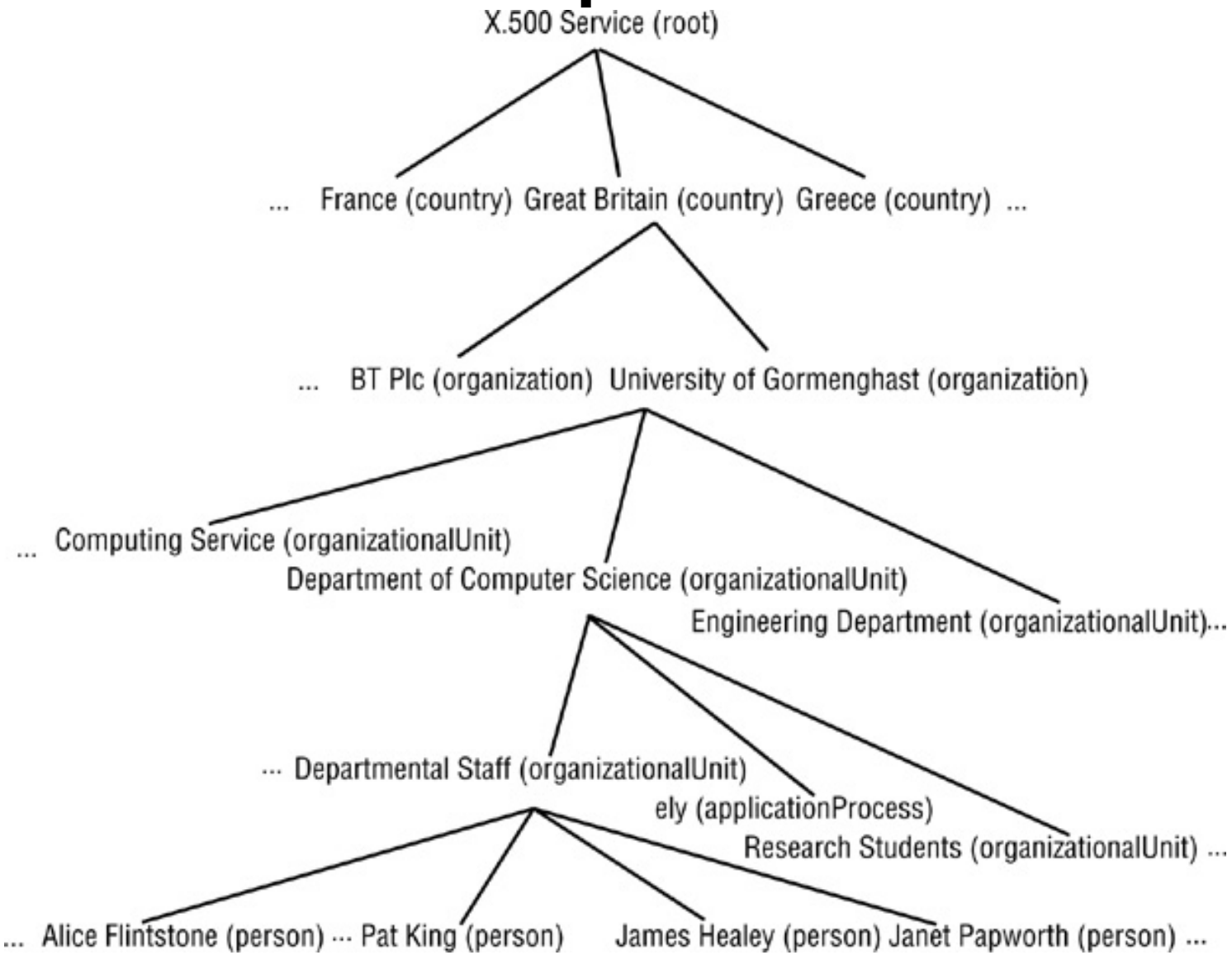
- X.500 is a directory service
  - Primary usage: descriptive queries
  - Can also be used as a name service
- Organized as a tree structure with names nodes
  - wide range of attributes stored at each node
  - search entries by any combination of attributes
- Directory Information Tree (DIT): X.500 name tree
- Directory Information Base (DIB): entire directory structure including the data associated with the nodes (partitioned on servers)

X.500 Service Architecture

- Servers: Directory Service Agents (DSA)
- Clients: Directory User Agents (DUA)
- Each DUA interacts with a single DSA that access other DSA servers to satisfy requests



Example DIT


X.500 Service (root)
... France (country) Great Britain (country) Greece (country) ...
... BT Plc (organization) University of Gormenghast (organization)
... Computing Service (organizationalUnit)
Department of Computer Science (organizationalUnit)
Engineering Department (organizationalUnit)...
... Departmental Staff (organizationalUnit)
ely (applicationProcess)
Research Students (organizationalUnit) ...
... Alice Flintstone (person) ... Pat King (person) James Healey (person) Janet Papworth (person) ...

Example DIB Entry

info	
	Alice Flintstone, Departmental Staff, Department of Computer Science, University of Gormenghast, GB
commonName	*uid*
Alice.L.Flintstone	alf
Alice.Flintstone	*mail*
Alice Flintstone	alf@dcs.gormenghast.ac.uk
A. Flintstone	Alice.Flintstone@dcs.gormenghast.ac.uk
surname	*roomNumber*
Flintstone	Z42
telephoneNumber	*userClass*
+44 986 33 4604	Research Fellow

- Note: data structure of DIB entries is very flexible (more attributes can be added)
- Distinguished Name (DN): attributes selected as distinguished

X.500 Access Requests

- **Read:**
  - Input: An absolute or relative name (a domain name in X.500 terminology) for an entry + list of attributes to read (or *)
  - Navigate the DIT, passing requests among DSA servers
  - Output: requested attributes
- **Search:** an attribute-based access request
  - Input: a base name (starting node in the DIT) and a filter expression (boolean expression to be evaluated for each node in the subtree rooted by base name)
  - Output: domain names in the subtree rooted by base name for which the filter evaluates to TRUE
- Example:
  - Find the commonNames of members of staff who occupy room Z42 in Department of CS at the University of Gormenghast
  - Read request to retrieve other attributes of the returned entries

Administration and updating of the DIB

- DSA interface: add, delete, and modify entries
- DIB partitioning: each organization will provide at least one server holding the details of the entities in that organization
- DIB partitions can be replicated in several servers

Thank You