Naming in Distributed Systems

Hong-Linh Truong
Distributed Systems Group,
Vienna University of Technology

truong@dsg.tuwien.ac.at
dsg.tuwien.ac.at/staff/truong

Learning Materials

- Main reading:
  - Tanenbaum & Van Steen, Distributed Systems: Principles and Paradigms, 2e, (c) 2007 Prentice-Hall
    - Chapter 5
- Others
  - George Coulouris, Jean Dollimore, Tim Kindberg, „Distributed Systems – Concepts and Design“, 2nd Edition
    - Chapter 9.
- Test the examples in the lecture

Outline

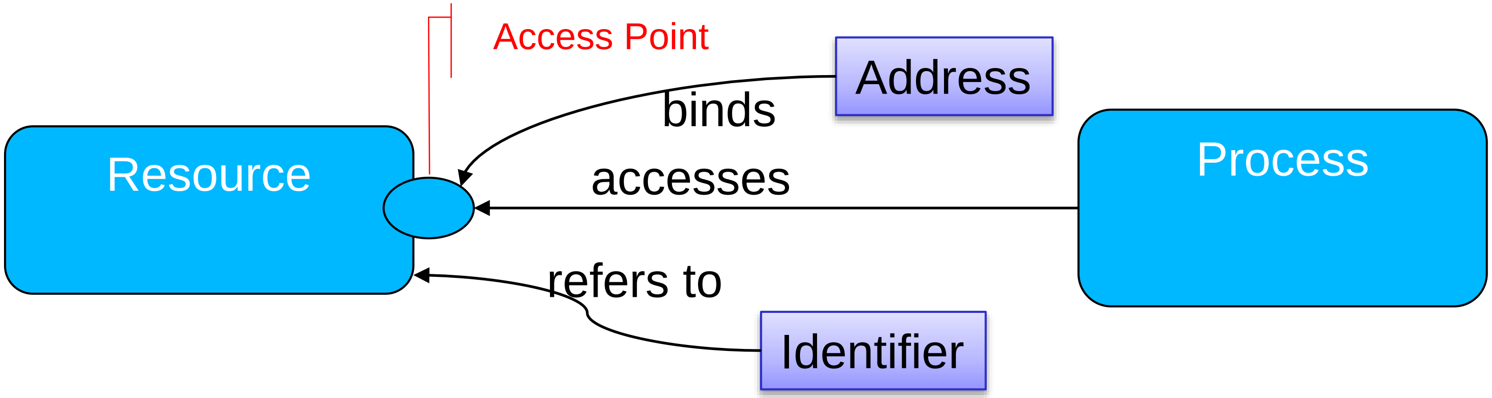
- Basic concepts and design principles
- Flat naming
- Structured naming
- Attribute-based naming
- Some naming systems in the Web
- Summary

BASIC CONCEPTS AND DESIGN PRINCIPLES

Why naming systems are important?

- Entity: any kind of objects we see in distributed systems: process, file, printer, host, communication endpoint, etc
- Diverse types of and complex dependencies among entities at different levels
  - E.g, printing service → the network level communication end points → the data link level communication end points
- But there are just so many entities, how do we create and manage names and identify an entity?

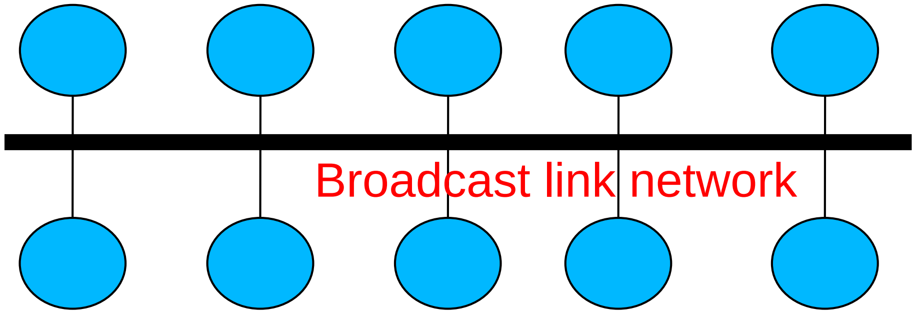
Names, identifiers, and addresses



- Name: set of bits/characters used to identify/refer to an entity, a collective of entities, etc. in a specific context or uniquely
    - Simply comparing two names, we might not be able to know if they refer to the same entity
- Identifier: a name that uniquely identifies an entity
    - the identifier is unique and refers to only one entity
- Address: the name of an access point

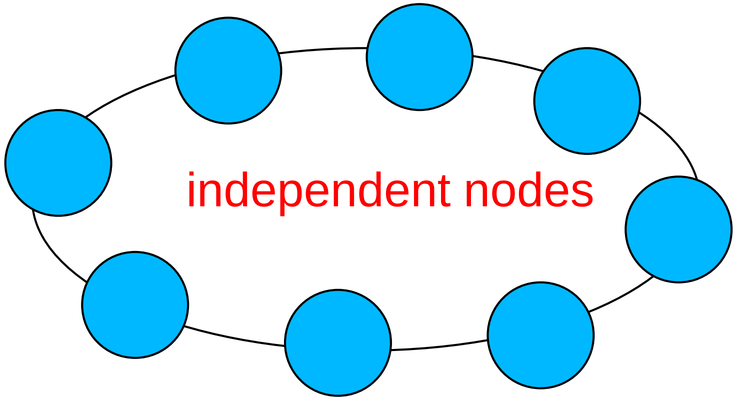
Naming design principles

- Naming design is based on specific system organizations and characteristics



Examples

- Network/Ethernet
- Identifier: IP and MAC address
- Name resolution: the network address to the data link address



- P2P systems
- Identifier: m-bit key
- Name resolution: distributed hash tables

Naming design principles

- Structures and characteristics of names are based on different purposes
- Data structure:
  - Can be simple, no structure at all, e.g., a set of bits:
    $ uuid
    bcff7102-3632-11e3-8d4a-0050b6590a3a
  - Can be complex
    - Include several data items to reflect different aspects on a single entity
  - Names can include location information/reference or not, e.g., GLN (Global Location Number) in logistics
  - Readability:
    - Human-readable or machine-processable formats

Naming design principles

- Diverse name-to-address binding mechanisms
    - How a name is associated with an address or how an identifier is associated with an entity
    - Names can be changed over the time and names are valid in specific contexts
        - Dynamic or static binding?
- Distributed or centralized management
    - Naming data is distributed over many places or not
- Discovery/Resolution protocol
    - Names are managed by distributed services
    - Noone/single system can have a complete view of all names

FLAT NAMING

Flat naming

Unstructured/flat names: identifiers have no structured description, e.g., just a set of bits

- Simple way to represent identifiers
- Do not contain information for locating the access point of the entity
- Examples
  - Internet Address at the Network layer
  - m-bit numbers in Distributed Hash Tables

Q1: Flat naming are suitable for which types of systems

Broadcast based Name Resolution

- Principles
  - Assume that we want find the access point of the entity en
  - Broadcast the identifier of en, e.g., broadcast(ID(en))
  - Only en will return the access point, when the broadcast message reaches nodes
- Examples
  - ARP: from IP address to MAC address (the datalink access point)

```
mail.infosys.tuwien.ac.at (128.131.172.240) at 00:19:b9:f2:07:55 [ether] on eth0
sw-ea-1.kom.tuwien.ac.at (128.131.172.1) at 00:08:e3:ff:fc:c8 [ether] on eth0
```

Dynamic systems

- Nodes form the system, no centralized coordination
- Nodes can join/leave/fail anytime
- A large number of nodes but a node knows only a subset of nodes
- Examples
  - Large-scale p2p systems, e.g., Chord, CAN (Content Addressable Network), and Pastry

Q1: How to define identifiers for such a system?

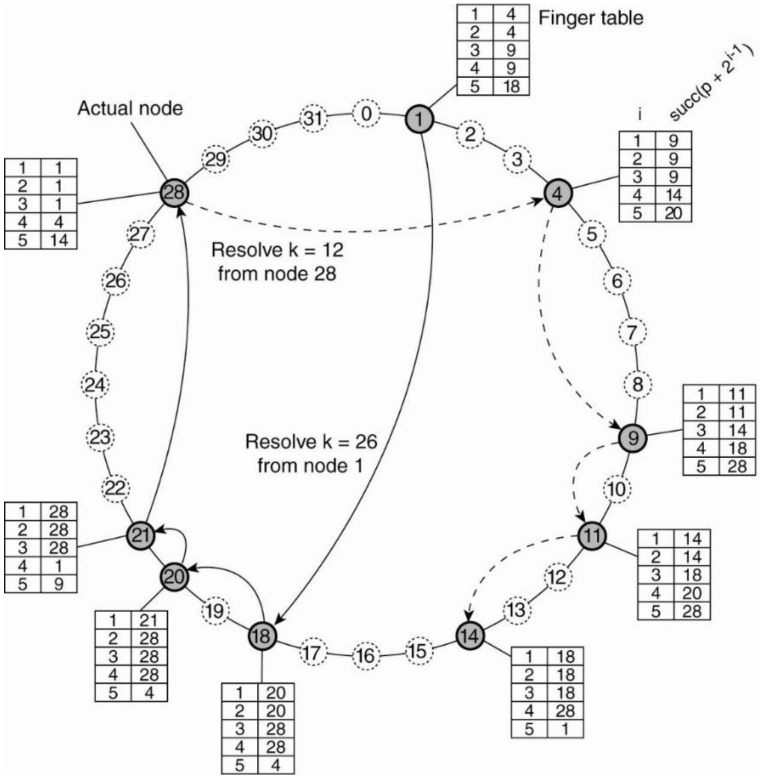
Distributed Hash Tables

- Main concepts
  - m-bit is used for the keyspace for identifiers
  - (Processing) Node identifier nodeID is one key in the keyspace
  - An entity en is identified by a hash function k=hash(en)
  - A node p is responsible for managing entities associated with a range of keys
    - If (k=hash(en) ∈ range(p)), then put (k, en) will store en in p
  - Nodes will relay messages (including entities/name resolution requests) till the messages reach the right destination

Q: Why DHT is useful for P2P systems? Is the nodeID fixed?

Example - Chord

- A ring network with $[0 \ldots 2^m - 1]$ positions among nodes in clockwise
- nodeID = hash(IP)
- the successor of k, successor(k), the smallest node identifier that ≥k.
- A key k of entity en will be managed by the first node p where p =successor(k)≥ k=hash(en)/the first node clockwise from k


Finger table
i
succ(p + 2^(i-1))
Actual node
Resolve k = 12 from node 28
Resolve k = 26 from node 1
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
Node 1: 1 4, 2 4, 3 9, 4 9, 5 18
Node 4: 1 9, 2 9, 3 9, 4 14, 5 20
Node 9: 1 11, 2 11, 3 14, 4 18, 5 28
Node 11: 1 14, 2 14, 3 18, 4 20, 5 28
Node 14: 1 18, 2 18, 3 18, 4 28, 5 1
Node 18: 1 20, 2 20, 3 28, 4 28, 5 4
Node 20: 1 21, 2 28, 3 28, 4 28, 5 4
Node 21: 1 28, 2 28, 3 28, 4 1, 5 9
Node 28: 1 1, 2 1, 3 1, 4 4, 5 14

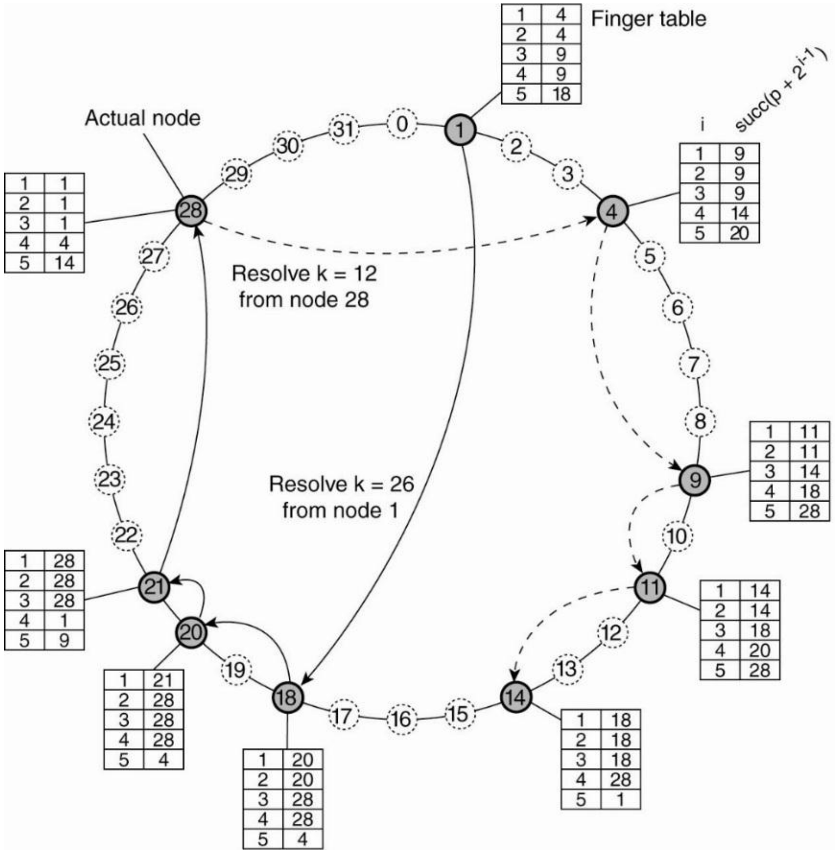
Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

http://pdos.csail.mit.edu/papers/chord:sigcomm01/

Example - Chord

- Resolving at p
  - Keep m entries in a finger table FT

$$FT_p[i]$$
$$= (successor(p$$



1	4
2	4
3	9
4	9
5	18

1	9
2	9
3	9
4	14
5	20

1	11
2	11
3	14
4	18
5	28

1	14
2	14
3	18
4	20
5	28

1	18
2	18
3	18
4	28
5	1

1	20
2	20
3	28
4	28
5	4

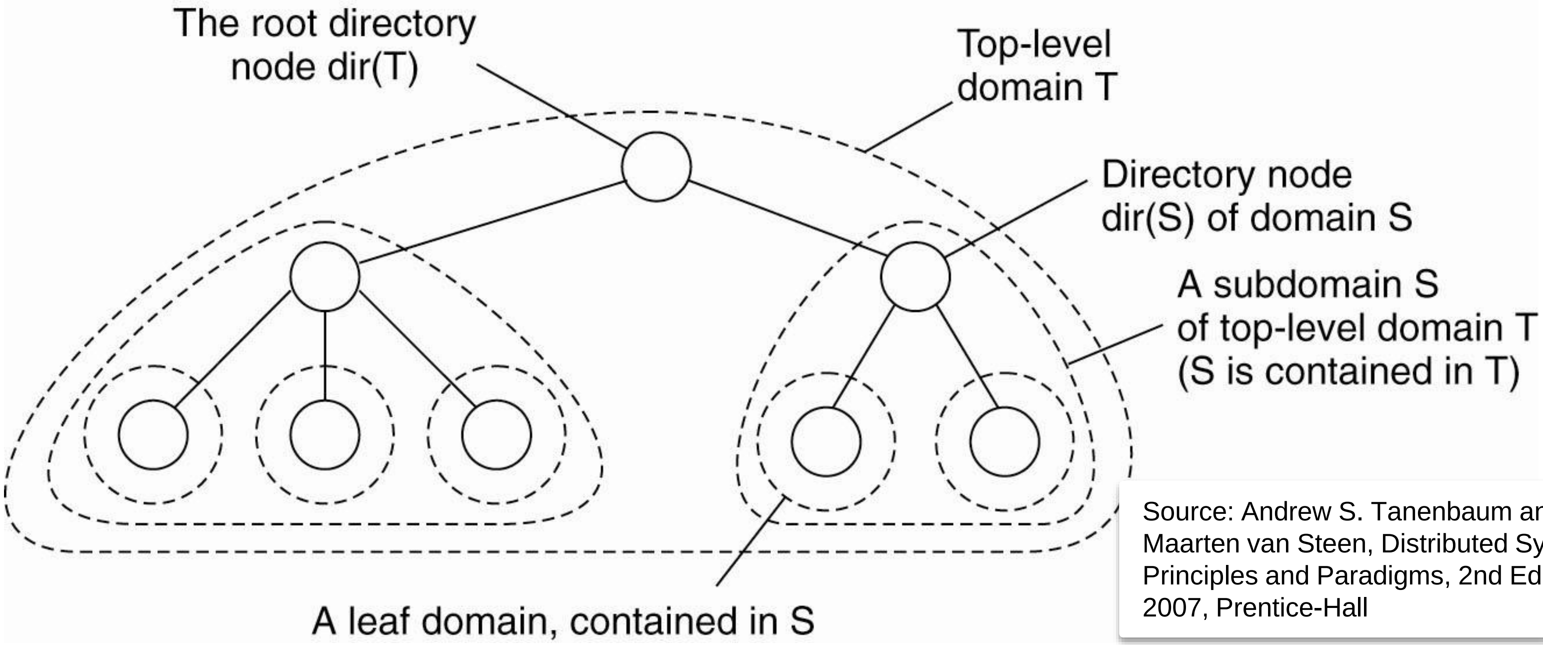
1	21
2	28
3	28
4	28
5	4

1	28
2	28
3	28
4	1
5	9

1	1
2	1
3	1
4	4
5	14

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

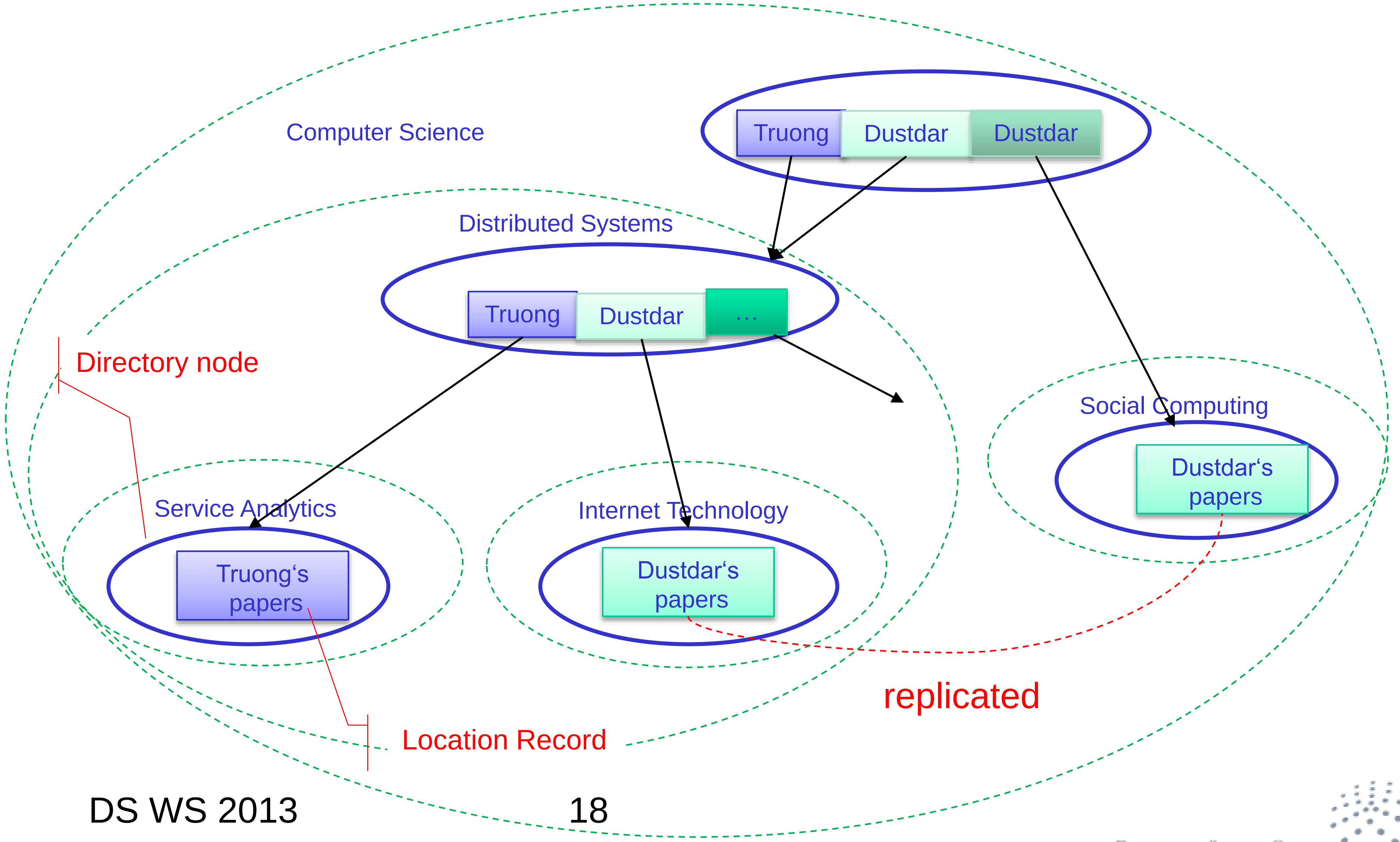
Name management and resolution - - Hierarchical approach


The root directory node dir(T)
Top-level domain T
Directory node dir(S) of domain S
A subdomain S of top-level domain T (S is contained in T)
A leaf domain, contained in S

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

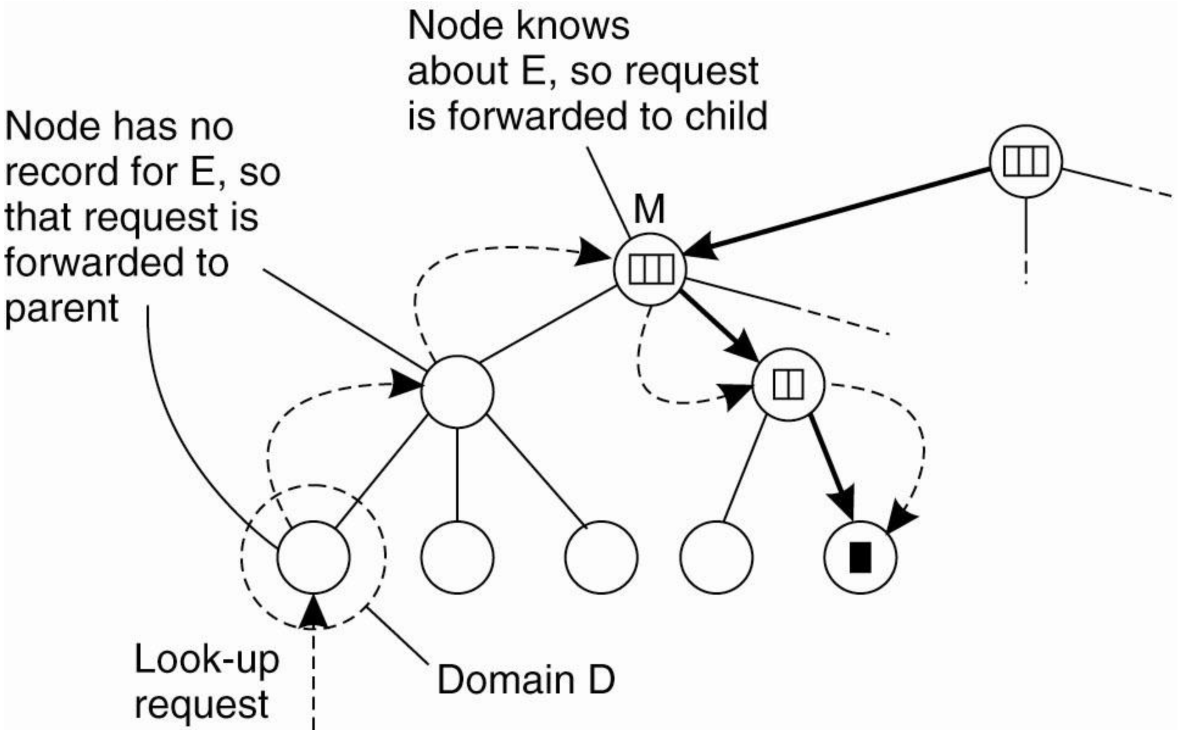
- The directory node has several location records.
- A location record is used to keep information about an entity in a domain D.
- The directory nodes contains both location records and pointers

Name management and resolution -- Hierarchical approach



Name management and resolution - - Hierarchical approach

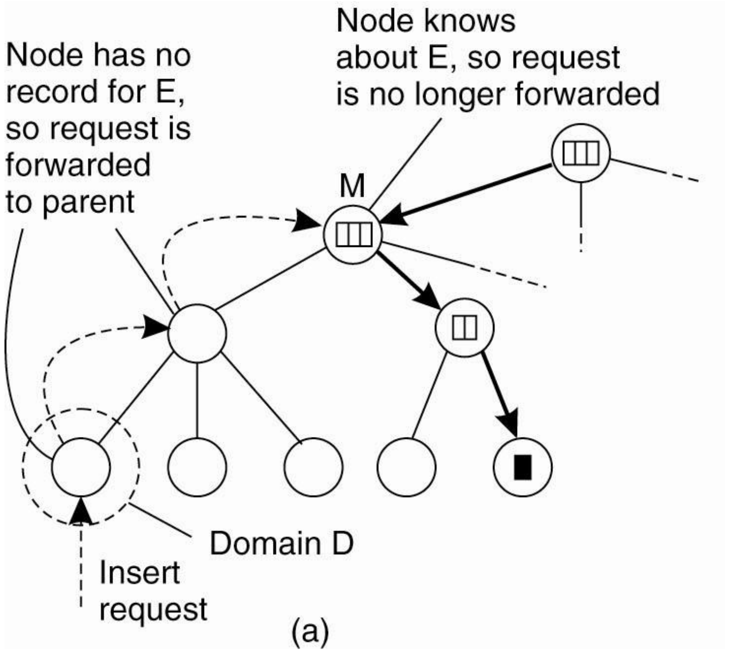
Lookup mechanism


Node knows about E, so request is forwarded to child
Node has no record for E, so that request is forwarded to parent
M
Look-up request
Domain D

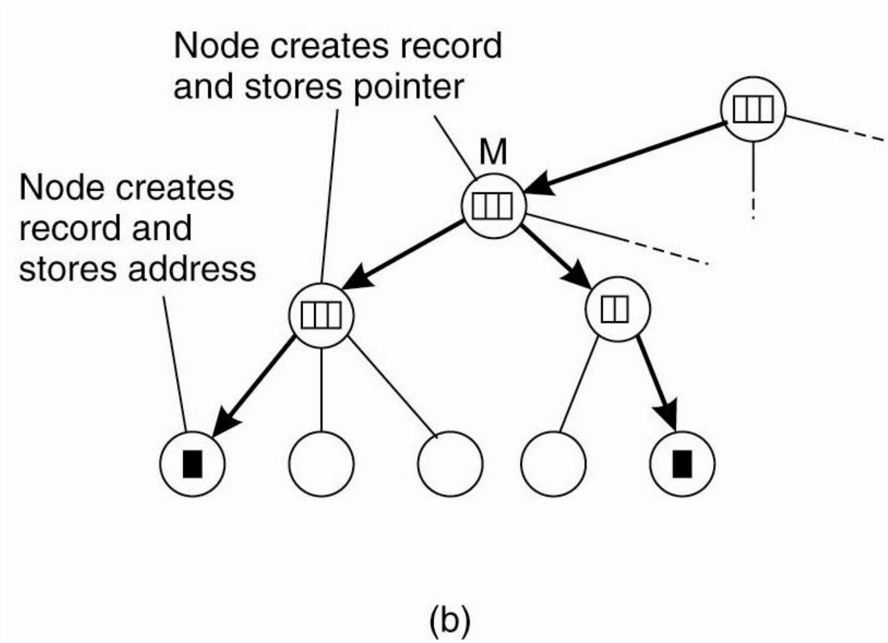
Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

Name management and resolution -- Hierarchical approach

Insert/update mechanism


Node has no record for E, so request is forwarded to parent
Node knows about E, so request is no longer forwarded
M
Domain D
Insert request
(a)

Insert request chain


Node creates record and stores pointer
M
Node creates record and stores address
(b)

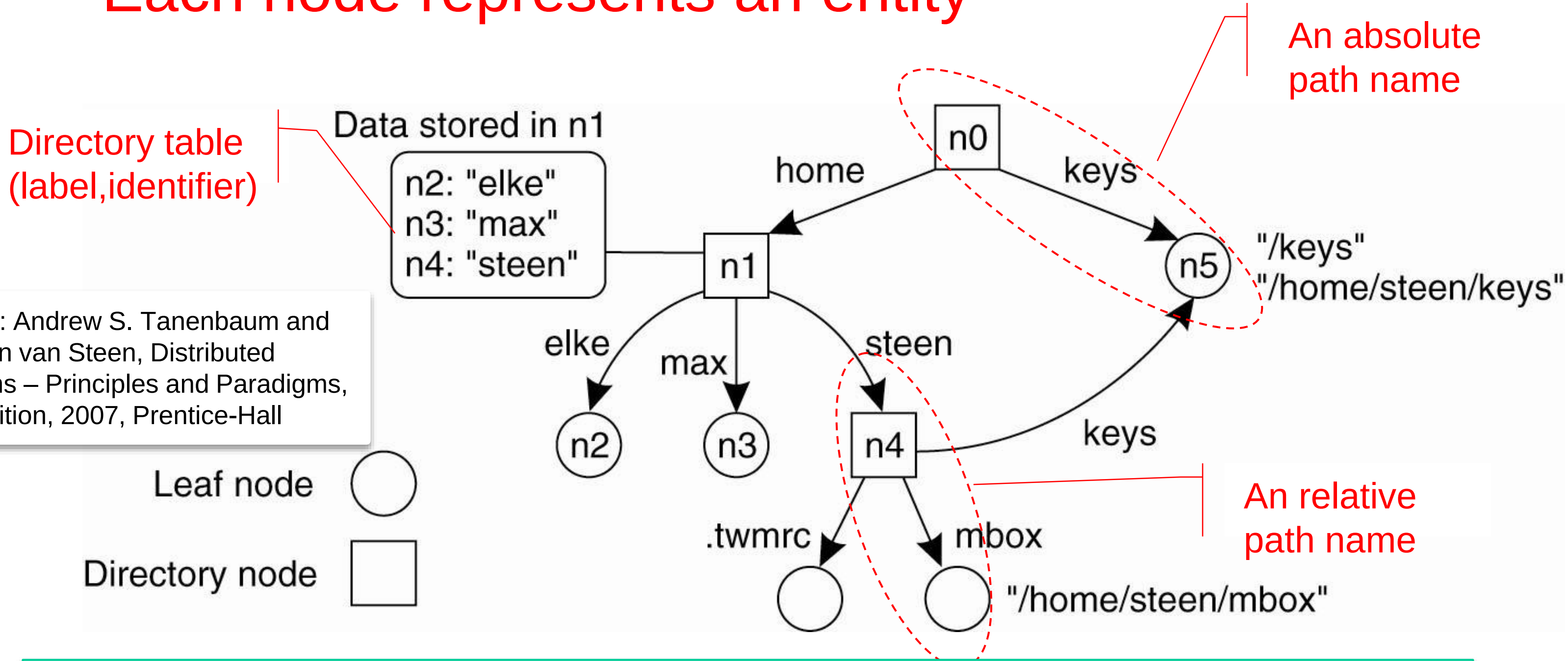
Create forwarding pointers chain

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

STRUCTURED NAMING

Name spaces

- Names are organized into a name space
- A name space can be modeled as a graph:
  - Leaf node versus directory node
- Each node represents an entity



Directory table (label,identifier)

An absolute path name

An relative path name

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

Q: How this differs from the flat naming with hierarchical approach?

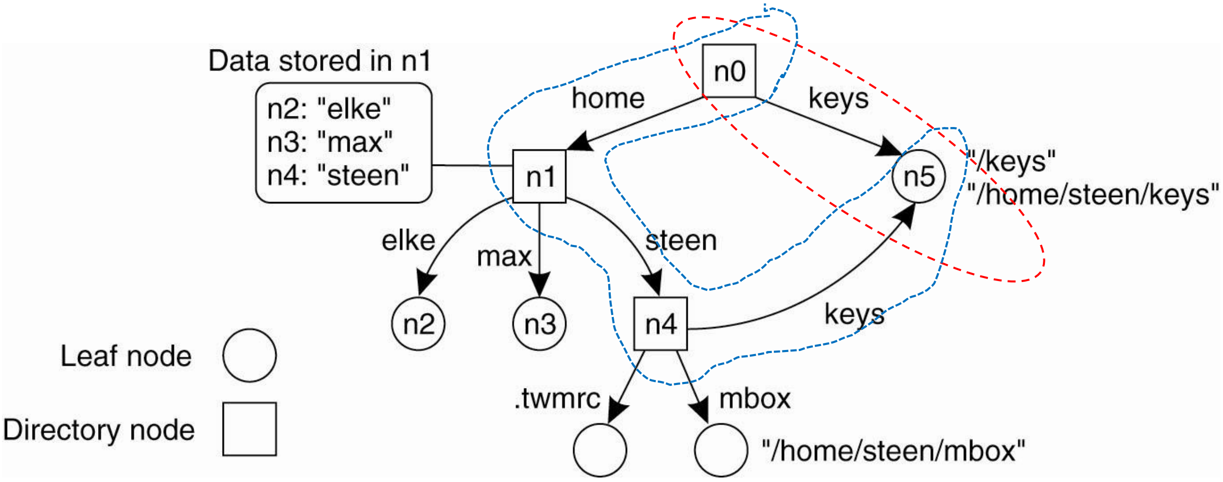
Name resolution – Closure Mechanism

- Name resolution: N:<label1,label2,label3,…labeln>
  - Start from node N
  - Lookup (label1,identifier1) in N‘s directory table
  - Lookup (label2, identifier2) in identifier1‘s directory table
  - and so on

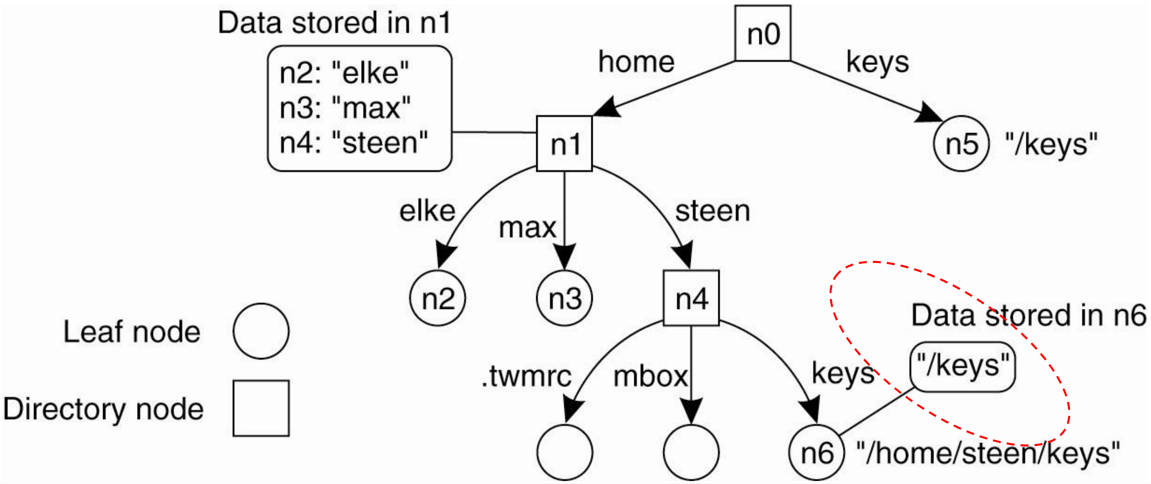
Closure Mechanism: determine where and how name resolution would be started

- E.g., name resolution for /home/truong/ds.txt ?
- Or for https://me.yahoo.com/a/.....

Enabling Alias Using Links



Hard links: multiple absolute paths names referring to the same node

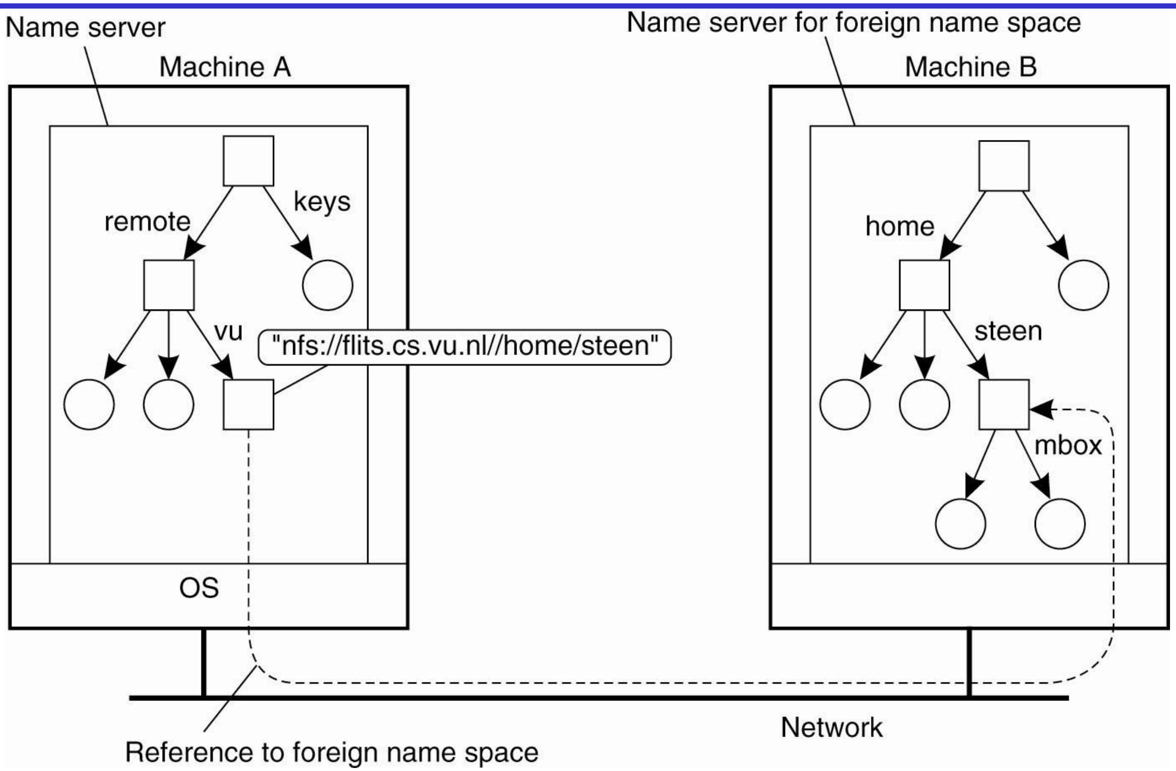


Symbolic links: leaf node storing an absolute path name

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

Name resolution - Mounting

- A directory node (mounting point) in a remote server can be mounted into a local node (mount point)



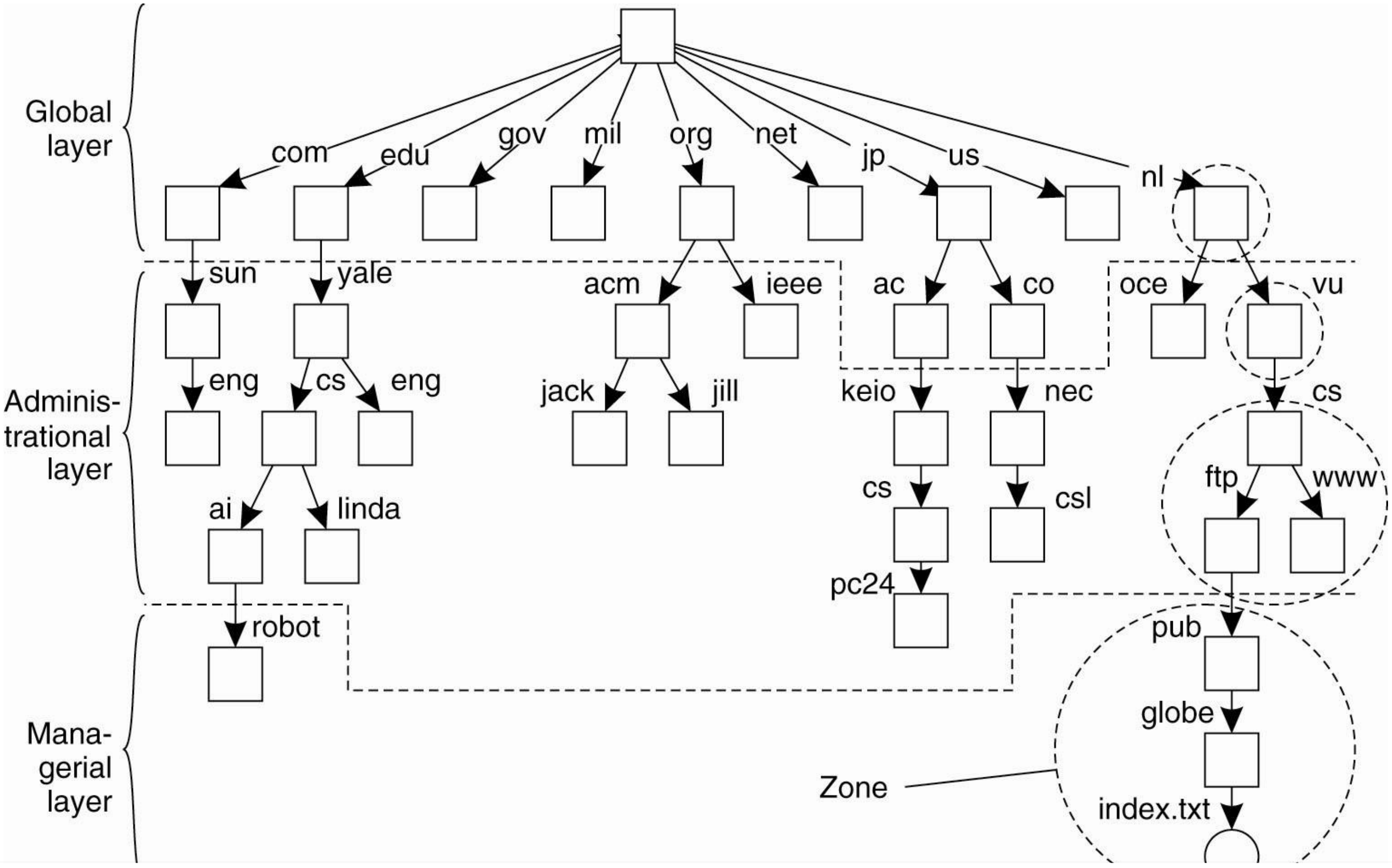
Name server
Machine A
Name server for foreign name space
Machine B
remote
keys
vu
"nfs://flits.cs.vu.nl//home/steen"
home
steen
mbox
OS
Network
Reference to foreign name space

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

Name space implementation

- Distributed name management
  - Several servers are used for managing names
- Many distribution layers
  - **Global layer:** the root node and its close nodes
  - **Administrational layer:** directory nodes managed within a single organization
  - **Managerial layer:** nodes typically change regularly.

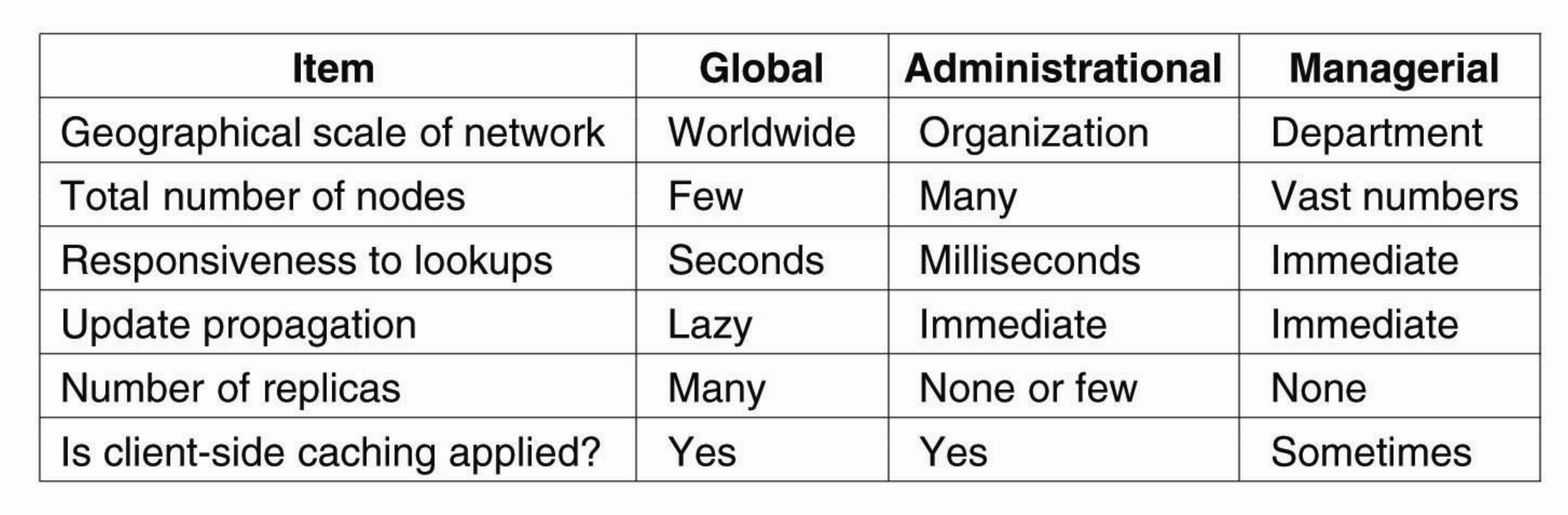
Example in Domain Name System



Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

aaa

Characteristics of distribution layers

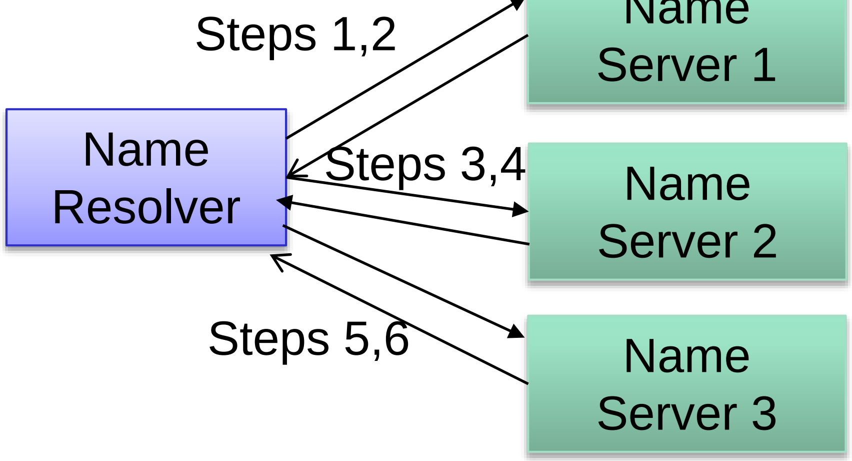


Item	Global	Administrational	Managerial
Geographical scale of network	Worldwide	Organization	Department
Total number of nodes	Few	Many	Vast numbers
Responsiveness to lookups	Seconds	Milliseconds	Immediate
Update propagation	Lazy	Immediate	Immediate
Number of replicas	Many	None or few	None
Is client-side caching applied?	Yes	Yes	Sometimes

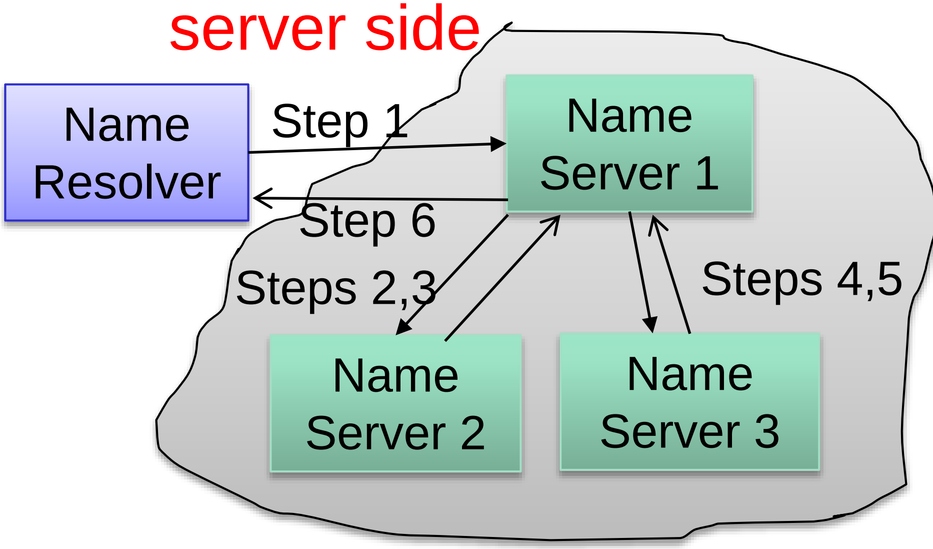
Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

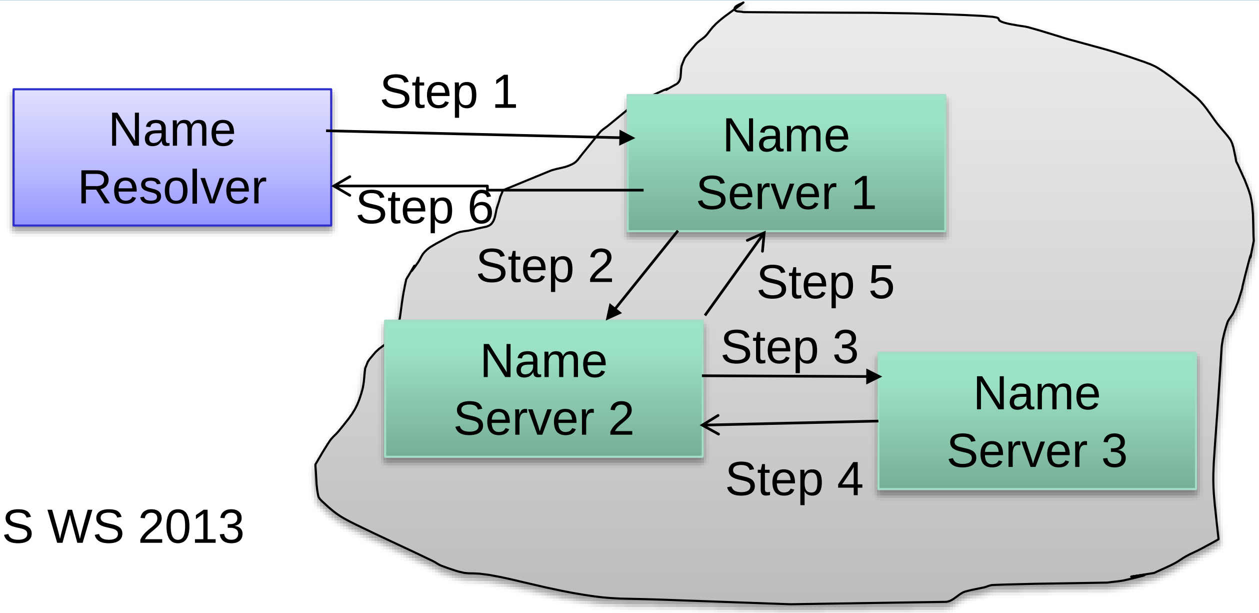
Name Resolution

Iterative name resolution at resolver side



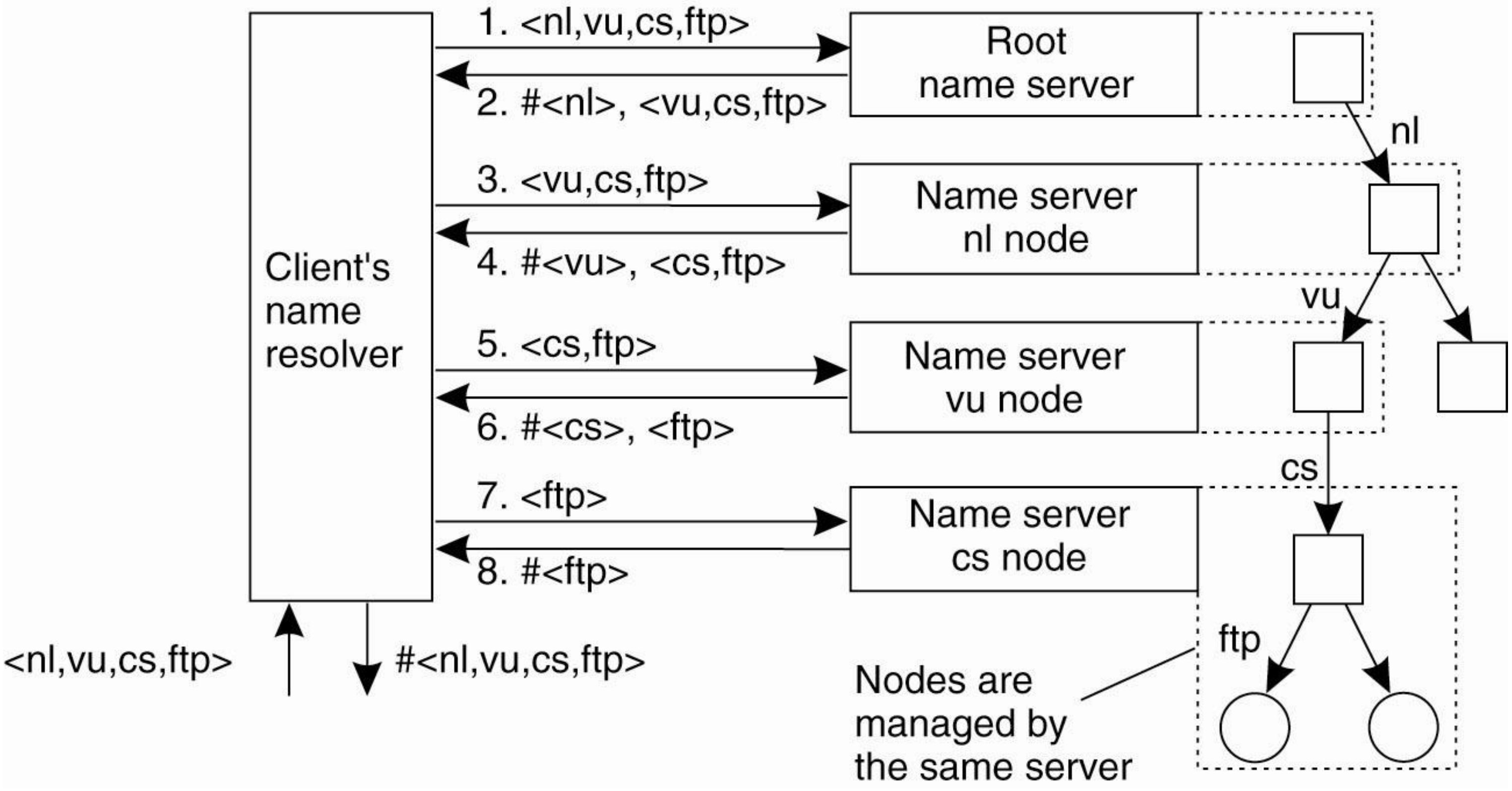
Iterative name resolution at server side





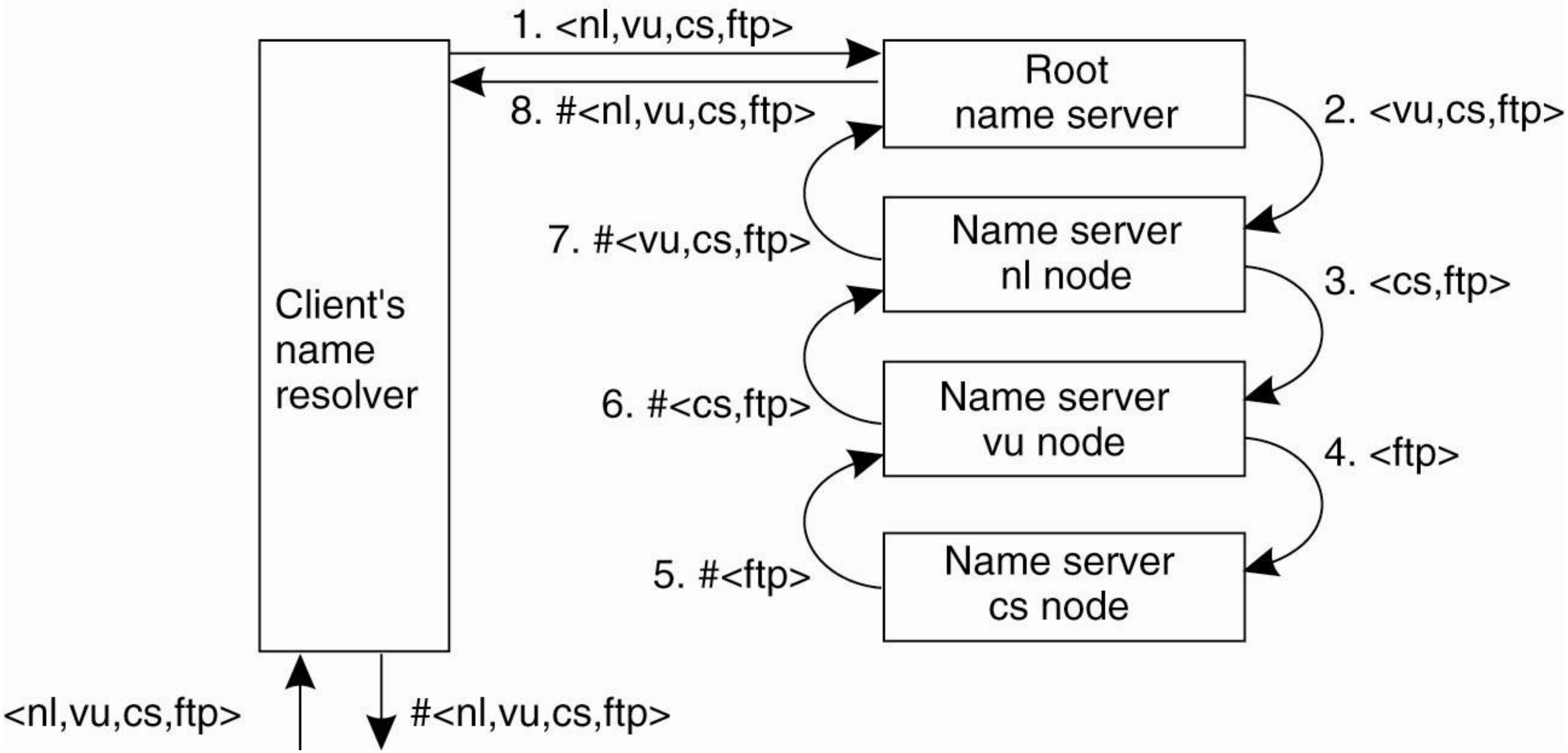
Recursive name resolution

Example -- Iterative name resolution



Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

Example -- Recursive name resolution



Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

Q: Pros and cons of recursive name resolution

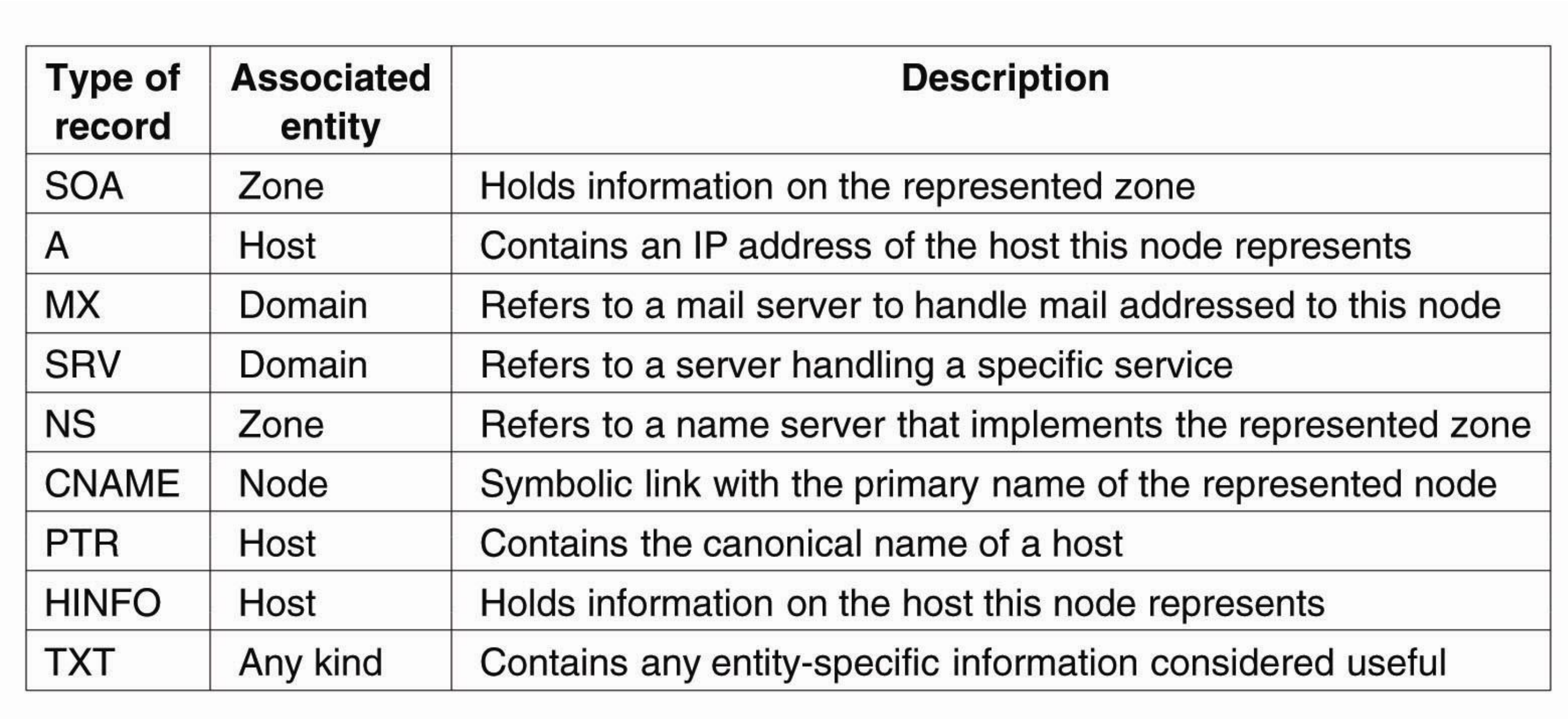
Domain Name System in Internet

- We use to remember „human-readable“ machine name → we have the name hierarchy
  - E.g., www.facebook.com
- But machines in Internet use IP address
  - E.g., 31.13.84.33
  - Application communication use IP addresses and ports
- DNS
  - Mapping from the domain name hierarchy to IP addresses

```
www.facebook.com      canonical name = star.c10r.facebook.com.
Name:    star.c10r.facebook.com
Address: 31.13.84.33
```

Domain Name System

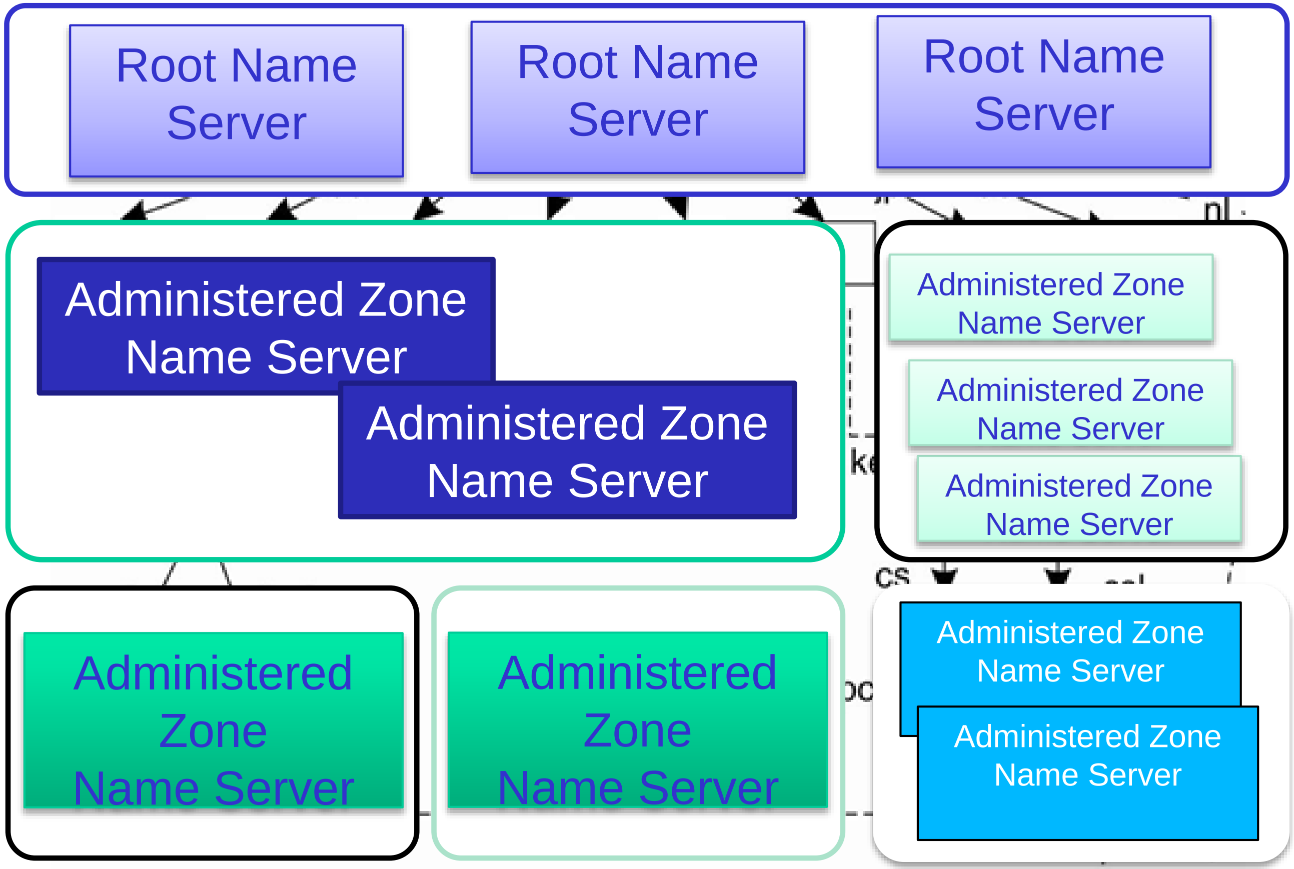
Information in records of DNS namespace



Type of record	Associated entity	Description
SOA	Zone	Holds information on the represented zone
A	Host	Contains an IP address of the host this node represents
MX	Domain	Refers to a mail server to handle mail addressed to this node
SRV	Domain	Refers to a server handling a specific service
NS	Zone	Refers to a name server that implements the represented zone
CNAME	Node	Symbolic link with the primary name of the represented node
PTR	Host	Contains the canonical name of a host
HINFO	Host	Holds information on the host this node represents
TXT	Any kind	Contains any entity-specific information considered useful

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

DNS Name Servers



Root Name Server

Root Name Server

Root Name Server

Administered Zone Name Server

Administered Zone Name Server

Administered Zone Name Server

Administered Zone Name Server

Administered Zone Name Server

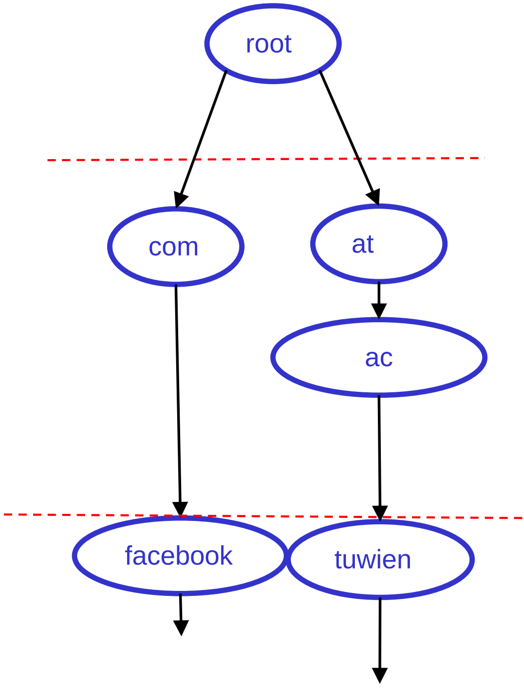
Administered Zone Name Server

Administered Zone Name Server

Administered Zone Name Server

Administered Zone Name Server

Example



root

com

at

ac

facebook

tuwien

- Authoritative name server: answer requests for a zone
- Primary and secondary servers: the main server and the replicated server (maintained copied data from the main server)
- Caching server

DNS Queries

- **Simple host name resolution**
  - Which is the IP of www.tuwien.ac.at?
- **Email server name resolution**
  - Which is the email server for truong@dsg.tuwien.ac.at ?
- **Reverse resolution**
  - From IP to hostname
- **Host information**
- **Other service**

Examples

- Iterative hostname resolution: http://www.simpledns.com/lookup-dg.aspx
- Mail server resolution: https://www.mailive.com/mxlookup/

ATTRIBUTE-BASED NAMING

Attributes/Values

- A tuple (attribute,value) can be used to describe a property
  - E.g., („country“,“Austria“), („language“, „German“),
- A set of tuples (attribute, value) can be used to describe an entity

AustriaInfo

Attribute	Value
CountryName	Austria
Language	German
MemberofEU	Yes
Capital	Vienna

Attribute-based naming systems

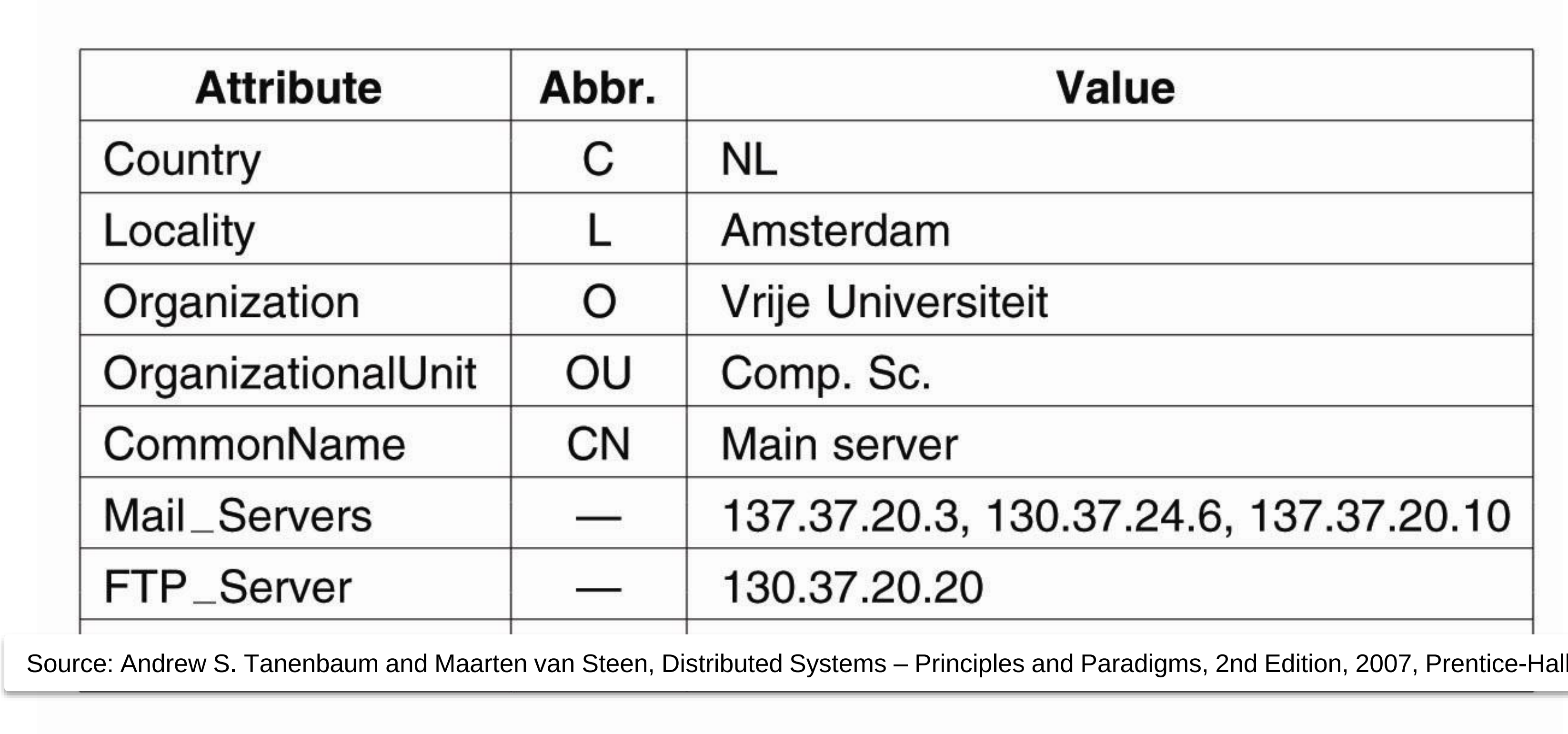
- Employ (attribute,value) tuples for describing entities
  - Why flat and structured naming are not enough?
- Also called directory services
- Naming resolution
  - Usually based on querying mechanism
  - Querying usually deal with the whole space
- Implementations
  - LDAP
  - RDF (Resource Description Framework)

LDAP data model

- Object class: describe information about objects/entities using **tuple(attribute,value)**
  - Hierarchical object class
- Directory entry: object entry for a particular object, alias entry for alternative naming and subentry for other information
- Directory Information Base (DIB): collection of all directory entries
  - Each entry is identified by a distinguished name (DN)
- Directory Information Tree (DIT): the tree structure for entries in DIB

LDAP – Lightweight Directory Access Protocol

- http://tools.ietf.org/html/rfc4510
- Example of attributes/values

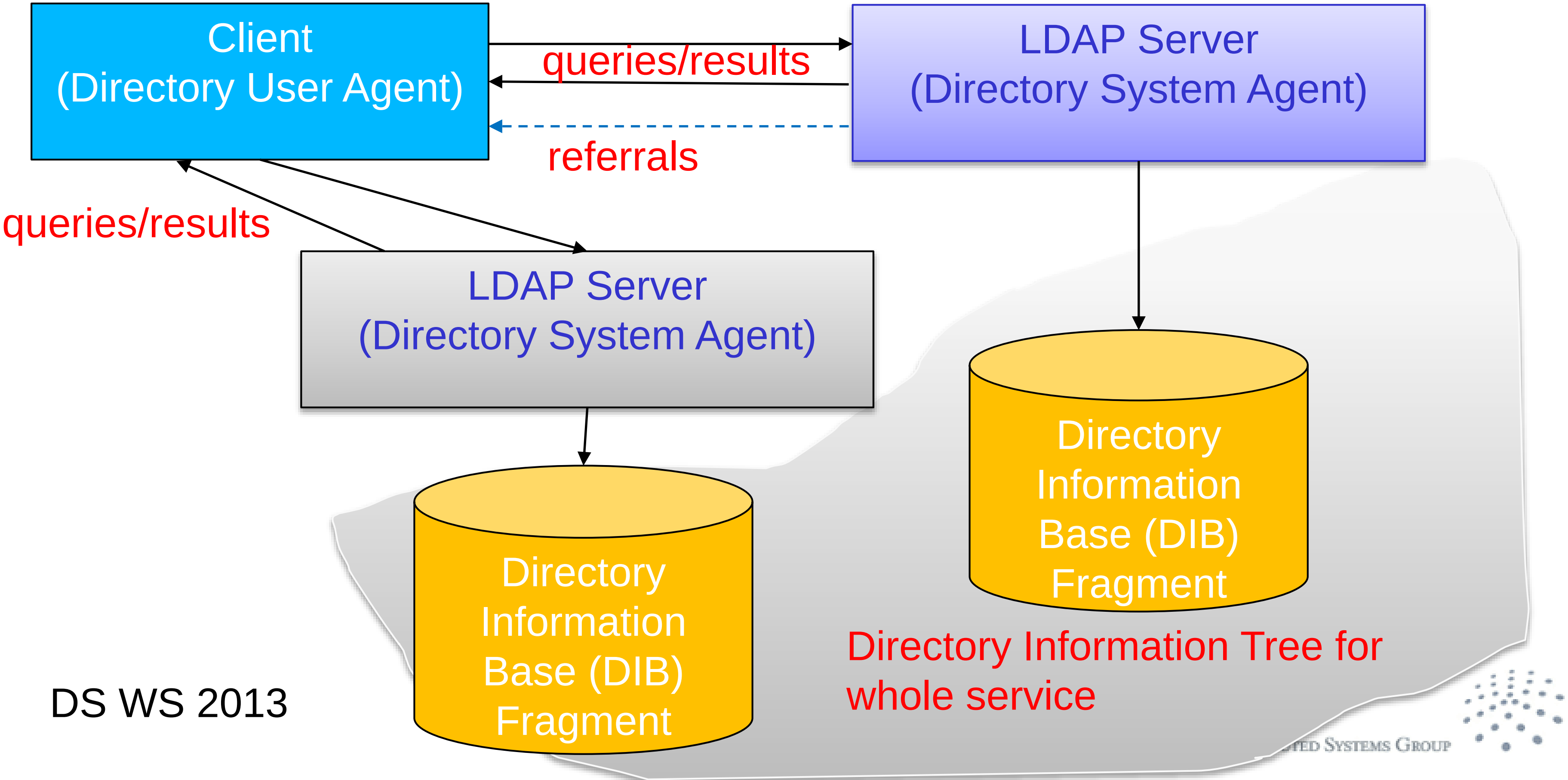


Attribute	Abbr.	Value
Country	C	NL
Locality	L	Amsterdam
Organization	O	Vrije Universiteit
OrganizationalUnit	OU	Comp. Sc.
CommonName	CN	Main server
Mail_Servers	—	137.37.20.3, 130.37.24.6, 137.37.20.10
FTP_Server	—	130.37.20.20

Source: Andrew S. Tanenbaum and Maarten van Steen, Distributed Systems – Principles and Paradigms, 2nd Edition, 2007, Prentice-Hall

LDAP-- Interaction

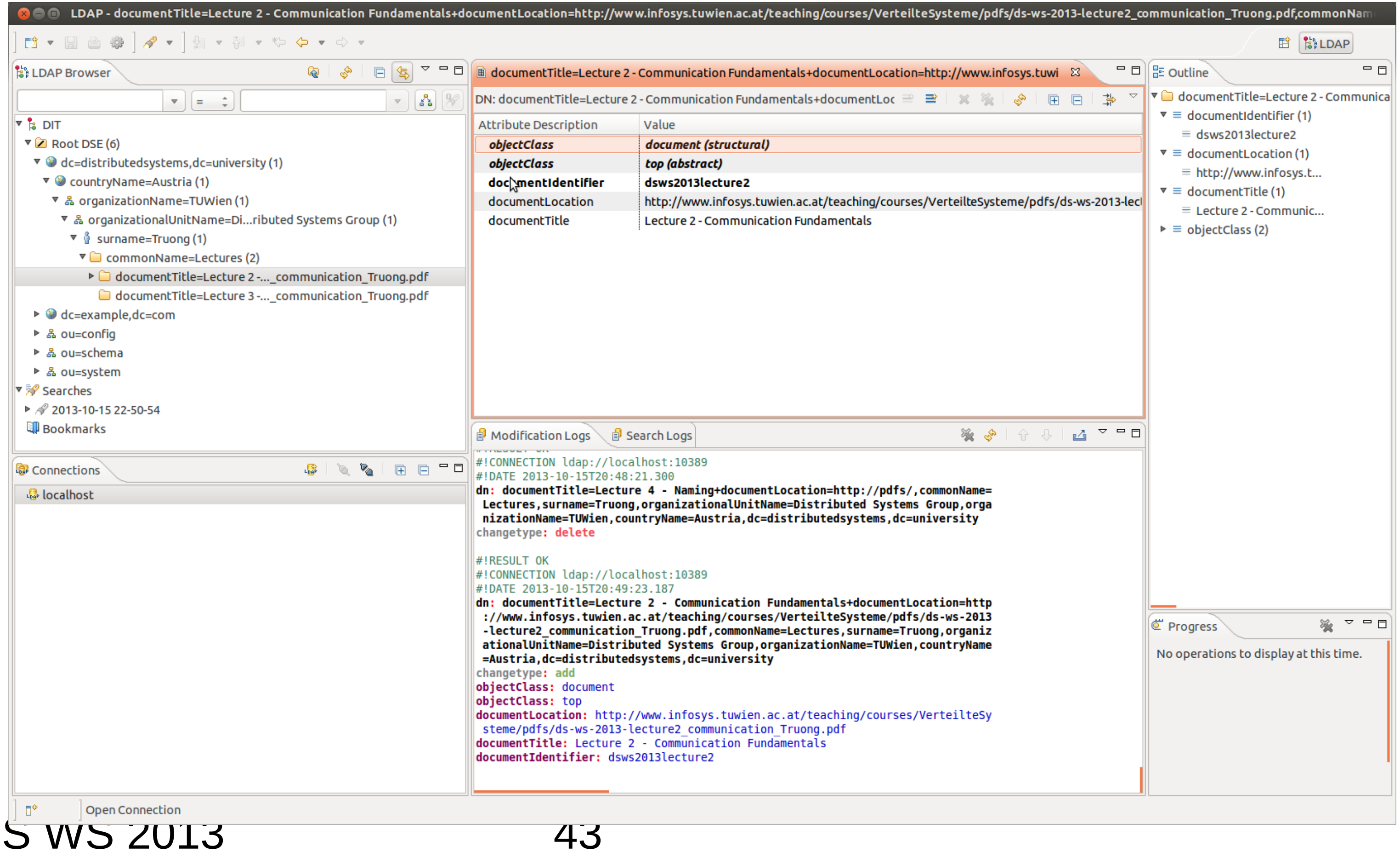
Client-server protocol



Directory Information Tree for whole service

Example with Apache DS/DS Studio

- http://directory.apache.org/
- Apache DS: a directory service supporting LDAP and others
- Apache Directory Studio: tooling platform for LDAP



LDAP - documentTitle=Lecture 2 - Communication Fundamentals+documentLocation=http://www.infosys.tuwien.ac.at/teaching/courses/VerteilteSysteme/pdfs/ds-ws-2013-lecture2_communication_Truong.pdf,commonNam

```
DIT
  Root DSE (6)
    dc=distributedsystems,dc=university (1)
      countryName=Austria (1)
        organizationName=TUWien (1)
          organizationalUnitName=Di...ributed Systems Group (1)
            surname=Truong (1)
              commonName=Lectures (2)
                documentTitle=Lecture 2 -..._communication_Truong.pdf
                documentTitle=Lecture 3 -..._communication_Truong.pdf
    dc=example,dc=com
    ou=config
    ou=schema
    ou=system
Searches
  2013-10-15 22-50-54
Bookmarks
```

documentTitle=Lecture 2 - Communication Fundamentals+documentLocation=http://www.infosys.tuwi

DN: documentTitle=Lecture 2 - Communication Fundamentals+documentLoc

Attribute Description	Value
objectClass	*document (structural)*
objectClass	*top (abstract)*
documentIdentifier	dsws2013lecture2
documentLocation	http://www.infosys.tuwien.ac.at/teaching/courses/VerteilteSysteme/pdfs/ds-ws-2013-lec
documentTitle	Lecture 2 - Communication Fundamentals

```
documentTitle=Lecture 2 - Communica
  documentIdentifier (1)
    dsws2013lecture2
  documentLocation (1)
    http://www.infosys.t...
  documentTitle (1)
    Lecture 2 - Communic...
  objectClass (2)
```

```
#!CONNECTION ldap://localhost:10389
#!DATE 2013-10-15T20:48:21.300
dn: documentTitle=Lecture 4 - Naming+documentLocation=http://pdfs/,commonName=
 Lectures,surname=Truong,organizationalUnitName=Distributed Systems Group,orga
 nizationName=TUWien,countryName=Austria,dc=distributedsystems,dc=university
changetype: delete

#!RESULT OK
#!CONNECTION ldap://localhost:10389
#!DATE 2013-10-15T20:49:23.187
dn: documentTitle=Lecture 2 - Communication Fundamentals+documentLocation=http
 ://www.infosys.tuwien.ac.at/teaching/courses/VerteilteSysteme/pdfs/ds-ws-2013
 -lecture2_communication_Truong.pdf,commonName=Lectures,surname=Truong,organiz
 ationalUnitName=Distributed Systems Group,organizationName=TUWien,countryName
 =Austria,dc=distributedsystems,dc=university
changetype: add
objectClass: document
objectClass: top
documentLocation: http://www.infosys.tuwien.ac.at/teaching/courses/VerteilteSy
 steme/pdfs/ds-ws-2013-lecture2_communication_Truong.pdf
documentTitle: Lecture 2 - Communication Fundamentals
documentIdentifier: dsws2013lecture2
```

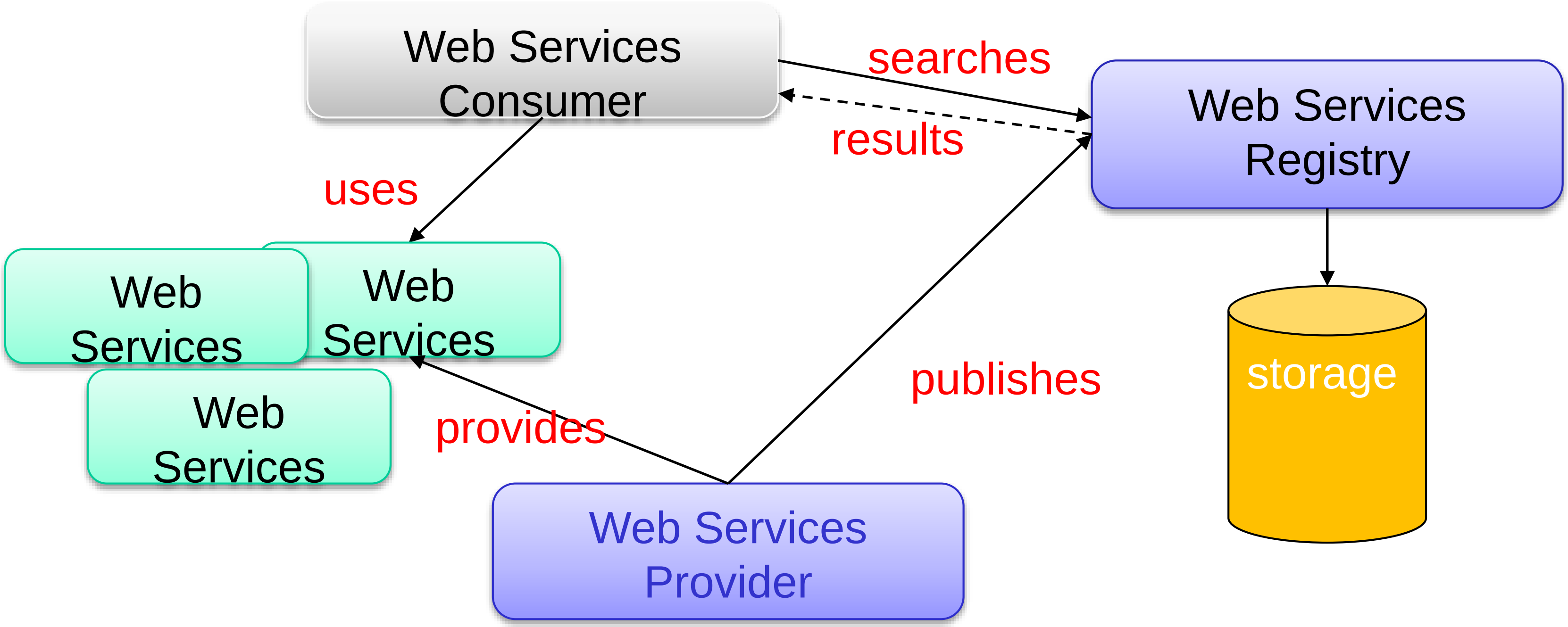
No operations to display at this time.

SOME NAMING SERVICES IN THE WEB

Web services – service identifier

- Web service: basically an entity which offers software function via well-defined, interoperable interfaces that can be accessed through the network
  - E.g., http://www.webservicex.net/globalweather.asmx
- Web services identifier:
  - A web service can be described via WSDL
  - Inside WSDL, there are several „addresses“ that identify where and how to call the service access points

Web services -- discovery



- Registry implementations
  - WSO2 Governance Registry - http://wso2.com/products/governance-registry/
  - java UDDI (jUDDI) - http://juddi.apache.org/

OpenID – people identifier in the Web

- Several services offering individual identifiers
  - Your google ID, Your yahoo ID, etc.
- But there will be no single provider for all people

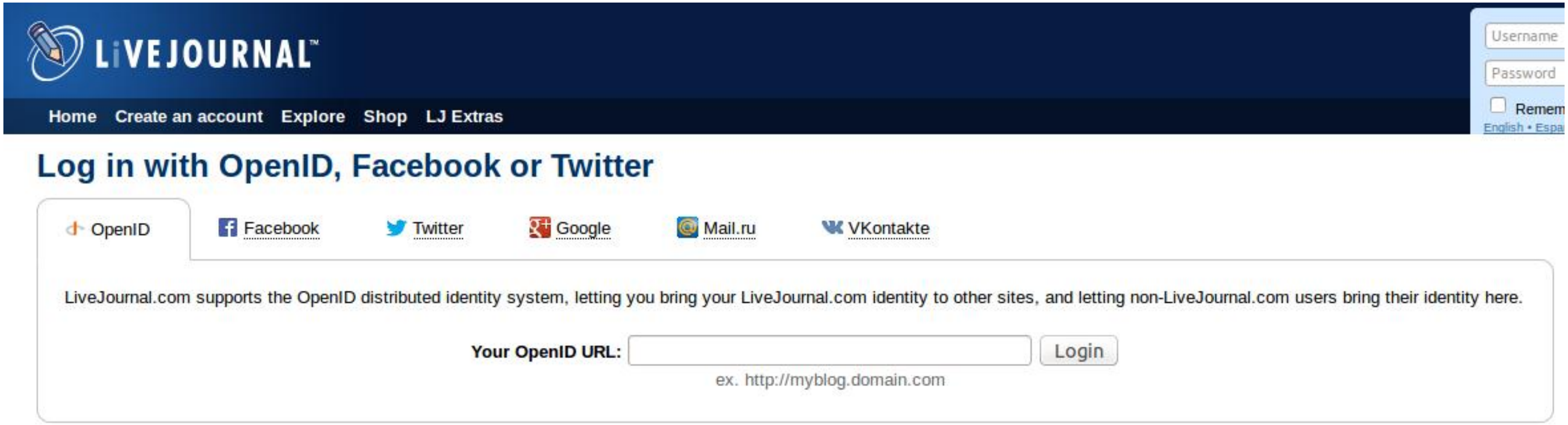
We need mechanisms to accept identifiers from different providers

- OpenID standard enables identifiers for people that can be accepted by several service provider
- An OpenID identifier is described as a URL
  - E.g., https://me.yahoo.com/a/.....

Q: Why OpenID identifier can be considered unique?

Example

Using OpenID to login to some services



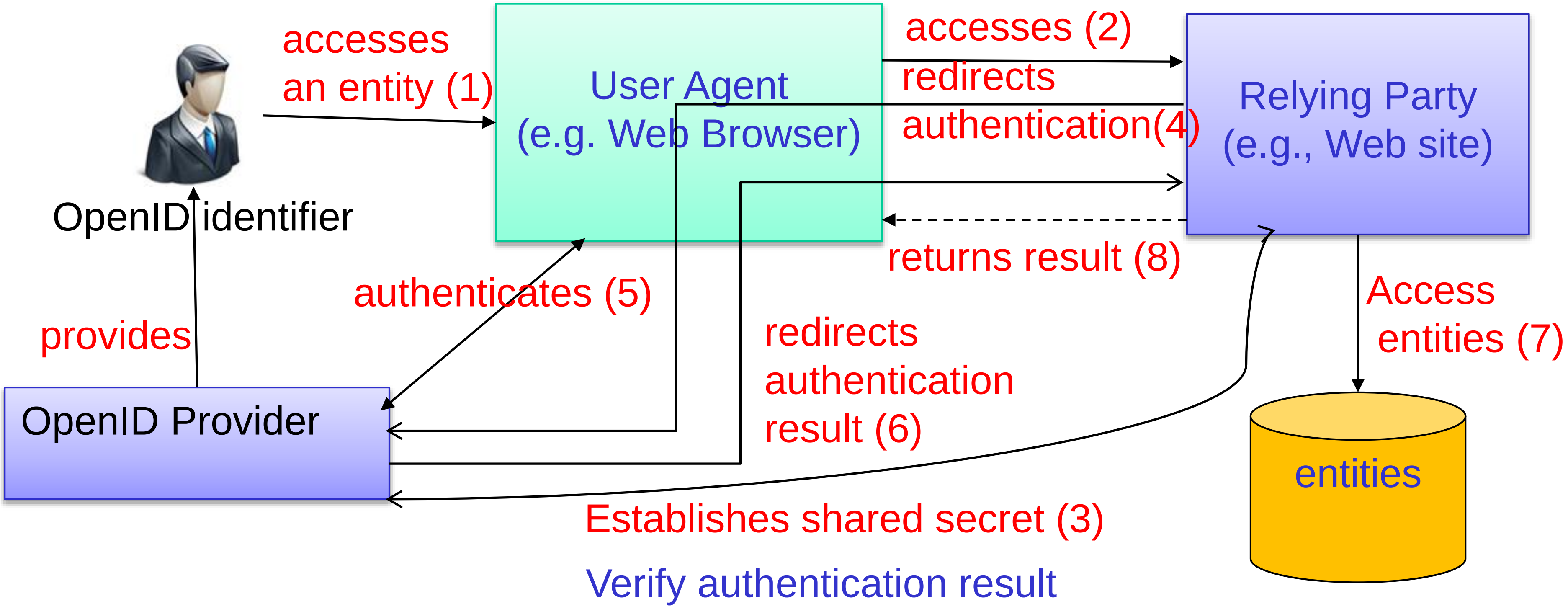
Log in with OpenID, Facebook or Twitter

LiveJournal.com supports the OpenID distributed identity system, letting you bring your LiveJournal.com identity to other sites, and letting non-LiveJournal.com users bring their identity here.

Your OpenID URL:

ex. http://myblog.domain.com

OpenID interactions



Summary

- Naming is a complex issue
  - Fundamental for other topics, e.g., communication and access control
- Different models
  - Flat, structured and attributed-based naming
- Different techniques to manage names
  - Centralized versus distributed
- Different protocols for naming resolution
- Dont forget to play some simple examples to understand existing concepts

Thanks for your attention

Hong-Linh Truong
Distributed Systems Group
Vienna University of Technology
truong@dsg.tuwien.ac.at
http://dsg.tuwien.ac.at/staff/truong