Naming in Distributed Systems

Distributed Systems
Sistemi Distribuiti

Andrea Omicini
andrea.omicini@unibo.it

Ingegneria Due
ALMA MATER STUDIORUM—Università di Bologna a Cesena

Academic Year 2010/2011

Outline

1. Basics
2. Names, Identifiers, Addresses
3. Flat & Structured Naming
4. Attribute-based Naming

These Slides Contain Material from [Tanenbaum and van Steen, 2007]

Slides were made kindly available by the authors of the book

- Such slides shortly introduced the topics developed in the book [Tanenbaum and van Steen, 2007] adopted here as the main book of the course
- Most of the material from those slides has been re-used in the following, and integrated with new material according to the personal view of the teacher of this course
- Every problem or mistake contained in these slides, however, should be attributed to the sole responsibility of the teacher of this course

Outline

1. Basics
2. Names, Identifiers, Addresses
3. Flat & Structured Naming
4. Attribute-based Naming

What is naming?

The issue of naming

- Mapping names onto computational entities
- E.g., *resources* in REST
- Finding the entity a name refers to is said *resolving* a name—*name resolution*
- *Naming system* – the portion of the system devoted to name resolution

The issue of naming in distributed systems

- Naming is an issue in computational systems in general
- Features of distributed system makes naming even more difficult
  - openness
  - location
  - mobility
  - distribution of the naming systems

The issue of naming system

- Distribution
- Scalability
- Efficiency

What is naming?

The issue of naming

- Mapping names onto computational entities
- E.g., *resources* in REST
- Finding the entity a name refers to is said *resolving* a name—*name resolution*
- *Naming system* – the portion of the system devoted to name resolution

The issue of naming in distributed systems

- Naming is an issue in computational systems in general
- Features of distributed system makes naming even more difficult
  - openness
  - location
  - mobility
  - distribution of the naming systems

The issue of naming system

- Distribution
- Scalability
- Efficiency

What is naming?

The issue of naming

- Mapping names onto computational entities
- E.g., *resources* in REST
- Finding the entity a name refers to is said *resolving* a name—*name resolution*
- *Naming system* – the portion of the system devoted to name resolution

The issue of naming in distributed systems

- Naming is an issue in computational systems in general
- Features of distributed system makes naming even more difficult
  - openness
  - location
  - mobility
  - distribution of the naming systems

The issue of naming system

- Distribution
- Scalability
- Efficiency

Outline

1 Basics

2 Names, Identifiers, Addresses

3 Flat & Structured Naming

4 Attribute-based Naming

Names

Defining a (distributed) naming system amounts at. . .

- defining a set of the *admissible names*
- defining the set of the *named entities*
- defining the association between names and entities

What is a name?

- A name is something that refers to an entity
- A string, a sequence of symbols, . . .
- Defining the set of the admissible names determines how we can speak about the system

Names

Defining a (distributed) naming system amounts at. . .

- defining a set of the *admissible names*
- defining the set of the *named entities*
- defining the association between names and entities

What is a name?

- A name is something that refers to an entity
- A string, a sequence of symbols, . . .
- Defining the set of the admissible names determines how we can speak about the system

Entities

Entities are to be used

- An entity is something one can operate on
- by accessing to it
- through an *access point*

Access point

- A special sort of entity in distributed systems
- used to access an entity
- like, the cell phone to access yourselves

Entities

Entities are to be used

- An entity is something one can operate on
- by accessing to it
- through an *access point*

Access point

- A special sort of entity in distributed systems
- used to access an entity
- like, the cell phone to access yourselves

Addresses

Accessing an entity thru an access point. . .

- requires an *address*
- like, your cell phone number
- In short, the address of an access point to an entity can be called the address of the entity

Can't we use addresses as names?

- They are names of some sort
- But, quite unfriendly for humans
- Location independence might be desirable

Addresses

Accessing an entity thru an access point. . .

- requires an *address*
- like, your cell phone number
- In short, the address of an access point to an entity can be called the address of the entity

Can't we use addresses as names?

- They are names of some sort
- But, quite unfriendly for humans
- Location independence might be desirable

Identifiers

Another type of name

1. An identifier refers to at most one entity
2. Each entity is referred to by at most one identifier
3. An identifier always refers to the same entity – it's never reused

Addresses vs. identifiers

- Identifiers are sorts of names
- But, different purposes
- E.g., while my user name `andrea.omicini` is not to be reused for another person of the Alma Mater (*identifier*), my cell number could instead be reused by someone else (*address*)

Identifiers

Another type of name

1. An identifier refers to at most one entity
2. Each entity is referred to by at most one identifier
3. An identifier always refers to the same entity – it's never reused

Addresses vs. identifiers

- Identifiers are sorts of names
- But, different purposes
- E.g., while my user name `andrea.omicini` is not to be reused for another person of the Alma Mater (*identifier*), my cell number could instead be reused by someone else (*address*)

Human-friendly Names

Identifiers and addresses are often in machine-readable form

- Humans cannot handle them easily
- This might create problems in the use, monitoring and control of distributed systems
- *Human-friendly names*

Resolving Names to Addresses

Main issue in naming

- How do we associate names and identifiers to addresses?
- In large, distributed, mobile, open systems, in particular?

Examples

- The simplest case: *name-to-address binding*, with a table of ⟨*name*, *address*⟩ pairs

← Problem: a centralised table does not work in large networks

- The DNS case: hierarchical composition

→ `www.apice.unibo.it` hierarchically resolved through a recursive lookup

Resolving Names to Addresses

Main issue in naming

- How do we associate names and identifiers to addresses?
- In large, distributed, mobile, open systems, in particular?

Examples

- The simplest case: *name-to-address binding*, with a table of $\langle name, address\rangle$ pairs

← Problem: a centralised table does not work in large networks

- The DNS case: hierarchical composition

→ `www.apice.unibo.it` hierarchically resolved through a recursive lookup

Outline

1. Basics

2. Names, Identifiers, Addresses

3. Flat & Structured Naming

4. Attribute-based Naming

Flat Naming

Basic Idea

- A name is just a flat sequence of chars / symbols
- Works in LANs

Examples

- *Broadcasting*: messages containing the identifier of the target entity is sent to everyone, only the machine containing the entity responds
- Example: ARP (Address Resolution Protocol)
- Problem: inefficient when the network grows
- *Multicasting*: only a restricted group of hosts receives the request
- Example: data-link level in Ethernet networks

Flat Naming

Basic Idea

- A name is just a flat sequence of chars / symbols
- Works in LANs

Examples

- *Broadcasting*: messages containing the identifier of the target entity is sent to everyone, only the machine containing the entity responds
- Example: ARP (Address Resolution Protocol)
- Problem: inefficient when the network grows
- *Multicasting*: only a restricted group of hosts receives the request
- Example: data-link level in Ethernet networks

Structured Naming

Basic Idea

- Flat names are good for machines, not for humans
- *Structured names* are composed by simple human-readable names – thus matching the natural limitations of human cognition

Example

- Internet name space

Structured Naming

Basic Idea

- Flat names are good for machines, not for humans
- *Structured names* are composed by simple human-readable names – thus matching the natural limitations of human cognition

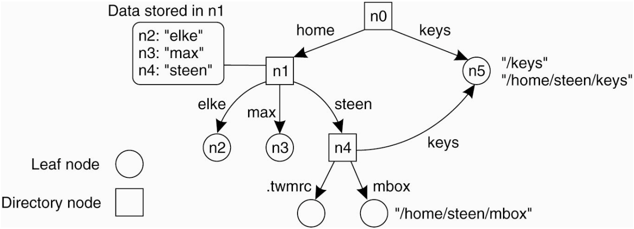
Example

- Internet name space

Name Spaces

Basic Idea

- Names are organised hierarchically, according to a labelled, directed graph – a *naming graph*
- *Leaf nodes* represent named entities
- *Directory nodes* have a number of outgoing edges, each labelled with an identifier



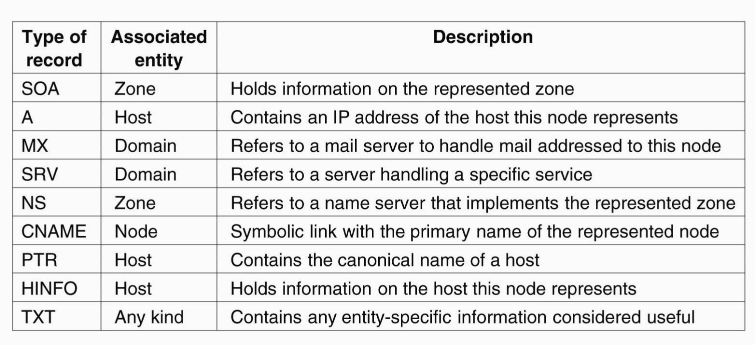
Symbolic link in a naming graph
[Tanenbaum and van Steen, 2007]

The Internet Domain Name Space (DNS)

The DNS Name Space

- Hierarchically organised as a rooted tree
- Each node (except root) has exactly one incoming edge, labelled with the name of the node
- A subtree is a *domain*
- A path name to its root node is a *path name*
- A node contains a collection of *resource records*

Resource Records



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

Most relevant types of resource records in a DNS node [Tanenbaum and van Steen, 2007]

Outline

1. Basics
2. Names, Identifiers, Addresses
3. Flat & Structured Naming
4. Attribute-based Naming

Limits of Flat & Structured Naming

Beyond Location Independence

- Flat naming allow for unique and location-independent way to refer to distributed entities
- Structured naming also provides for human-friendliness
- However, distributed systems are more and more information-based – information could also be the basis for looking for an entity
- Exploiting information associated to entities to locate them

Attribute-based Naming

Description as pairs

- Many way to describe an entity could be used
- Most popular: a collection of $\langle attribute, value\rangle$ pairs associated to an entity to describe it
- *Attribute-based naming*

A.k.a. Directory services

- Attribute-based naming systems are also known as *directory services*
- The essential point: choosing the right set of attributes to describe resources

Attribute-based Naming

Description as pairs

- Many way to describe an entity could be used
- Most popular: a collection of $\langle attribute, value \rangle$ pairs associated to an entity to describe it
- *Attribute-based naming*

A.k.a. Directory services

- Attribute-based naming systems are also known as *directory services*
- The essential point: choosing the right set of attributes to describe resources

Resource Description Framework (RDF)

RDF: A model for describing resources

- Each resource is a triplet $\langle \mathit{Subject}, \mathit{Predicate}, \mathit{Object} \rangle$
- E.g., $\langle \mathit{FootballPlayer}, \mathit{fullname}, \text{"}\mathit{Gaby\ \ Mudingay}\text{"} \rangle$ describe a resource *FootballPlayer* whose *fullname* is "*Gaby Mudingay*"
- Each subject, predicate, object can be a resource itself
- References in RDF are essentially URLs

Querying a directory service

- If resource description are stored, they can be queried when looking for a certain resource
- The query could return a reference to the resource, to be fetched by the application

Resource Description Framework (RDF)

RDF: A model for describing resources

- Each resource is a triplet $\langle Subject, Predicate, Object\rangle$
- E.g., $\langle FootballPlayer, fullname, "Gaby\ \ Mudingay"\rangle$ describe a resource *FootballPlayer* whose *fullname* is *"Gaby Mudingay"*
- Each subject, predicate, object can be a resource itself
- References in RDF are essentially URLs

Querying a directory service

- If resource description are stored, they can be queried when looking for a certain resource
- The query could return a reference to the resource, to be fetched by the application

Hierarchical Implementations I

Combining structured & attribute-based naming

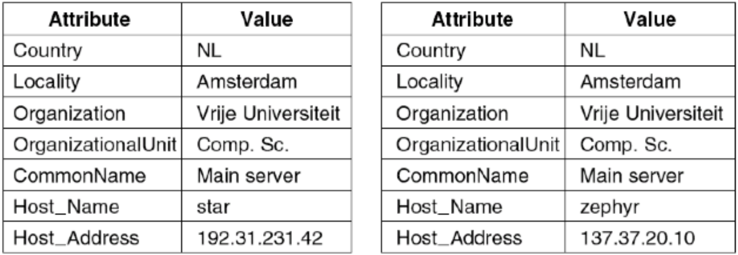
- Distributed directory services
  - Lightweight Directory Access Protocol (LDAP)
  - Example: MS Active Directory

Hierarchical Implementations II

Hierarchy through LDAP attribute-based names

- An LDAP directory service contains a number of *directory entries* – a collection of ⟨*attribute*, *value*⟩ pairs, similar to DNS resource records
- The directory entries in an LDAP directory service constitute the *directory information base* (DIB)—there, each record is uniquely named
- Naming attributes are called Relative Distinguished Names (RDN)—they are combined to form a globally-unique name, which is a structured name
- As a result, the *Directory Information Tree* (DIT) is a collection of directory entries forming the naming graph of an LDAP directory

Hierarchical Implementations III



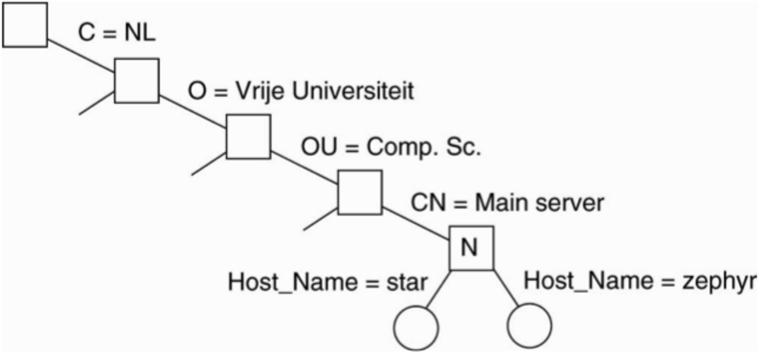
Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Comp. Sc.
CommonName	Main server
Host_Name	star
Host_Address	192.31.231.42

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Comp. Sc.
CommonName	Main server
Host_Name	zephyr
Host_Address	137.37.20.10

(b)

Two LDAP directory entries with hierarchical naming. . .
[Tanenbaum and van Steen, 2007]

Hierarchical Implementations IV



. . . along with the corresponding (partial) DIT [Tanenbaum and van Steen, 2007]

Summing Up

Naming is a general issue

- Particularly relevant in the distributed setting
- Different approaches to naming are possible: flat, structured, attribute-based
- Typically, naming systems take a hybrid stance to the naming problem
- DNS and LDAP are paradigmatic examples of naming systems

References I

Tanenbaum, A. S. and van Steen, M. (2007).
Distributed Systems. Principles and Paradigms.
Pearson Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition.

Naming in Distributed Systems

Distributed Systems
Sistemi Distribuiti

Andrea Omicini
andrea.omicini@unibo.it

Ingegneria Due
ALMA MATER STUDIORUM—Università di Bologna a Cesena

Academic Year 2010/2011