Distributed Systems

Seminar 7 – Naming and Directory Services

Dr. Ing. Thomas Springer
Technische Universität Dresden
Chair of Computer Networks
Explain the main functions of naming and directory services. What is the main difference between the two services?

- **Naming services** carry out name interpretations, by which logical names are mapped to addresses. In this way, logical server names for example can be mapped to concrete server instances, and as the case may be, object references. A direct binding of resources can thereby be avoided, whereby a flexible assignment of users to resources is possible. Main functions thereby are the registration of pairs of names and addresses as well as the search for logical names, whereby the corresponding addresses are returned.
  - Simple name interpretation, mapping between names and addresses
  - Example: DNS, RMI Registry

- **Directory services** are an extension of naming services. In addition to the basic functions of naming services, directory services enable an assignment of attributes to logical names so that resources can be discovered not only by means of logical names but also by means of particular properties.
  - attribute based search and ranking
  - Example: X.500, LDAP, UDDI
Different servers must communicate with each other in order to carry out a distributed name interpretation. What possibilities exist for the realization of this and what are the associated consequences for the querying client?

The procedures of **Chaining** and **Referral** can be distinguished for the realisation of distributed name interpretations.

Synonym: Chaining=recursive, Referral=iterative

With **Chaining**, the resolution of names is carried out internally by the distributed name servers and takes place therewith transparently for the client. The client need only to send a name query to a known name server, then receive the result from this.

With **Referral** and in contrast to Chaining, the client must send queries to all name servers involved in the name resolution and receives only a part of the result from each server, or as the case may be a reference to another server. The complexity of the name resolution lies thereby with the client. However with this solution the name servers are more decoupled and the client can undertake an optimisation of the name resolution.
Dr. Thomas Springer

DS Seminar - 7. Naming and Directory Services

Chaining=recursive

Referral=iterative
Two German companies “Computer” and “Network” have corresponding departments “Development”, “Sales” and “Research”.

a) Present in a tree-like diagram, a hierarchical namespace for these two companies using the DNS name schema.

b) Add the English company “Commercial” with the departments “Development”, “Service” and “Research” for DNS.

c) Provide an example of a relative name within the company “Computer” for DNS. In which context is this name interpreted?
- Relative names: development, sales, research
- Naming context:
  - DNS: development.computer.de
From the execution of the command `dig +trace -t A www.scss.tcd.ie` the following result was printed. Sketch the name resolution of an DNS query based that information. Check the result of the name resolution by using the `nslookup` command.

Lookup IPs with:
- `root-servers.org`
- `https://www.huque.com/app/dnsstat/detail/ie/`
- `http://www.tcpiputils.com/browse/ip-address/134.226.53.22`
Domain name to be resolved: www.scss.tcd.ie

Recursive query

Client

configured name server
141.30.66.135

Root name server
a.root-servers.net

ie name server
d.ns.ie

tcd.ie name server
auth-ns3.tcd.ie

scss name server
ns2.scss.tcd.ie

Iterative queries

1: request
2: request
3: ie
4: request
5: tcd.ie
6: request
7: scss.tcd.ie
8: request
9: 134.226.56.2
10: 134.226.56.2

Dr. Thomas Springer
DS Seminar - 7. Naming and Directory Services
The German companies “Computer” and “Network” have the departments “Development”, “Sales” and “Research”, the English company “Commercial” has the departments “Administration” and “Service”

a) Look up the definition of the object classes in RFC 4519. Propose the structure of a Directory Information Tree that can store the given information according to the X.500 standard.

b) Provide an example of a relative name within the company “Computer” for X.500. In which context is this name interpreted?

c) Which object class could be used to store the employees of the two companies as substructure of the department they belong to. The following information should be stored for each employee: name, address, telephone number, email and employee type.

d) Every employee who is working for the company for more than 10 years gets assigned the status of a senior. Name the search query for listing all senior employees across all organizations in Germany.
3.2. 'country'

The 'country' object class definition is the basis of an entry that represents a country.
(Source: X.521 [X.521])

( 2.5.6.2 NAME 'country'
    SUP top
    STRUCTURAL
    MUST c
    MAY ( searchGuide $
        description ) )

3.8. 'organization'

The 'organization' object class is the basis of an entry that represents a structured group of people.
(Source: X.521 [X.521])

( 2.5.6.4 NAME 'organization'
   SUP top
   STRUCTURAL
   MUST o
   MAY ( userPassword $ searchGuide $ seeAlso $
       businessCategory $ x121Address $ registeredAddress $
       destinationIndicator $ preferredDeliveryMethod $
       telexNumber $ teletexTerminalIdentifier $
       telephoneNumber $ internationalISDNNumber $
       facsimileTelephoneNumber $ street $ postOfficeBox $
       postalCode $ postalAddress $ physicalDeliveryOfficeName $
       st $ I $ description ) )
3.11. 'organizationalUnit'

The 'organizationalUnit' object class is the basis of an entry that represents a piece of an organization.
(Source: X.521 [X.521])

( 2.5.6.5 NAME 'organizationalUnit'
   SUP top
   STRUCTURAL
   MUST ou
   MAY ( businessCategory $ description $ destinationIndicator $
   facsimileTelephoneNumber $ internationalISDNNumber $ l $
   physicalDeliveryOfficeName $ postalAddress $ postalCode $
   postOfficeBox $ preferredDeliveryMethod $
   registeredAddress $ searchGuide $ seeAlso $ st $ street $
   telephoneNumber $ teletexTerminalIdentifier $
   telexNumber $ userPassword $ x121Address ) )
- Relative names: development, sales, research
- Naming context:
  - X.500: /.../C=DE/O=Computer/OU=Development
  - LDAP: ou=Development,o=Computer,c=DE, dc=seminarDS,dc=com

3.3. 'dcObject'
The 'dcObject' object class permits an entry to contains domain component information. This object class is defined as auxiliary, because it will be used in conjunction with an existing structural object class.

```
( 1.3.6.1.4.1.1466.344 NAME 'dcObject'
  SUP top
  AUXILIARY
  MUST dc )
```

The 'dc' ('domainComponent' in RFC 1274) attribute type is a string holding one component, a label, of a DNS domain name naming a host.
3.12 'person'

The 'person' object class is the basis of an entry that represents a human being.

( 2.5.6.6 NAME 'person'
  SUP top
  STRUCTURAL
  MUST ( sn $
       cn )$
  MAY ( userPassword $
       telephoneNumber $
       seeAlso $ description ) )
The following information should be stored for each employee:
• name, address, telephone number, email and employee type.

3.9. 'organizationalPerson'

The 'organizationalPerson' object class is the basis of an entry that represents a person in relation to an organization.

( 2.5.6.7 NAME 'organizationalPerson'
   SUP person
   STRUCTURAL
   MAY ( title $ x121Address $ registeredAddress $
       destinationIndicator $ preferredDeliveryMethod $
       telexNumber $ teletexTerminalIdentifier $
       telephoneNumber $ internationalISDNNumber $
       facsimileTelephoneNumber $ street $ postOfficeBox $
       postalCode $ postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l ) )
**inetOrgPerson** defined in RFC2798: Internet Organizational Person

The inetOrgPerson represents people who are associated with an organization in some way. It is a structural class and is derived from the organizationalPerson class.

```
( 2.16.840.1.113730.3.2.2
  NAME 'inetOrgPerson'
  SUP organizationalPerson
  STRUCTURAL
  MAY ( audio $ businessCategory $ carLicense $ departmentNumber $
    displayName $ employeeNumber $ employeeType $ givenName $
    homePhone $ homePostalAddress $ initials $ jpegPhoto $
    labeledURI $ mail $ manager $ mobile $ o $ pager $ photo $ roomNumber $ secretary $ uid $ userCertificate $
    x500uniqueIdentifier $ preferredLanguage $ userSMIMECertificate $ userPKCS12
  )
)```
ldapsearch -H ldap://localhost:10389 -x -D "uid=admin,ou=system" -W -b "dc=seminarDS,dc=com" -s sub -a always -z 1000 "(employeeType=senior)" "objectClass"

-H Specify URI(s) referring to the ldap server(s)
-x Use simple authentication
-D binddn Use the Distinguished Name binddn to bind to the LDAP directory.
-W Prompt for simple authentication.
-b Use searchbase as the starting point for the search instead of the default.
-s base|one|sub Specify the scope of the search to be one of base, one, or sub to specify a base object, one-level, or subtree search. The default is sub.
-a never|always|search|find Specify how aliases dereferencing is done. Should be one of never, always, search, or find to specify that aliases are never dereferenced, always dereferenced, dereferenced when searching, or dereferenced only when locating the base object for the search. The default is to never dereference aliases.
-z retrieve at most sizelimit entries for a search.