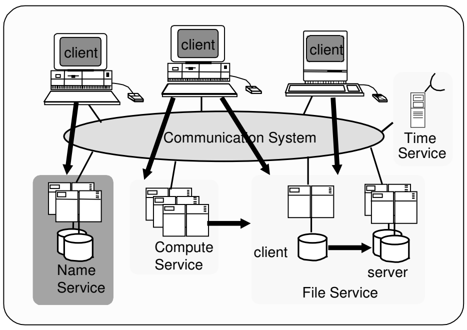
06-06798 Distributed Systems

Lecture 8: Name Services

Overview

- Naming concepts
  - name space, contexts, hierarchies
- The service
  - function and goals
  - name resolution
  - replication and caching
- Examples
  - Domain Name Service (DNS)
  - Jini discovery service

Distributed Service



Naming concepts

Names = strings used to identify objects (files, computers, people, processes, objects).

- Textual names (human readable)
  - used to identify individual services, people
    - email address: xxx@cs.bham.ac.uk
    - URL: www.cdk3.net
  - or groups of objects
    - multicast address (e.g. IP Multicast, group of hosts)
    - broadcast address (e.g. Ethernet, all hosts)

Naming concepts ctd

- Numeric addresses (location dependent)
  - 147.188.195.11
- Object identifiers
  - pure names (=bit patterns), usually numeric and large
  - never reused (include timestamp)
  - location independent
  - used for identification purposes

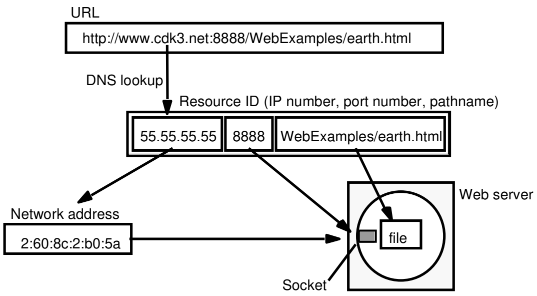
No real distinction between names and addresses.

Both must be looked up to obtain lower-level data (= name resolution).

Examples of name services

- DNS (=Domain Name Service)
  - maps domain names to IP addresses
- GNS (=Global Name Service)
  - maps global names to their attributes
  - scalable, can handle change
- X500 directory service
  - maps person's name to email address, phone number
- Jini discovery service
  - looks up objects according to attributes

DNS names & look-ups



Name space

- Name space = collection of all valid names recognised by a service with
  - a syntax for specifying names, and
  - rules for resolving names (left to right, etc)
- Naming context = maps a name on to primitive attributes directly, or on to another context and derived name (usually by prefixing)
  - telephone no: country, area, number
  - Internet host names: contexts=domains
  - Unix file system: contexts=directories

Name space ctd

- Name binding
  - an association between a name and an object
  - names bound to attributes, one of which may be address
- Naming domain
  - has authority that assigns names to objects within a name space or context
    - SoCS assigns login names
  - object may be registered more than once within context
- Multiple names
  - alias (alternative name for an object)
  - symbolic name (alternative name which maps to a path name in the name space)

Hierarchic name spaces

- Sequence of name tokens resolved in different context
  - syntax: name token (text string) + delimiter
  - DNS: cs.bham.ac.uk
  - Unix: /usr/bin
- Structure reflects organisational structure
  - name changes if object migrates
  - names relative to context or absolute
  - local contexts managed in a distributed fashion
- Examples
  - domain names, Unix file system, etc

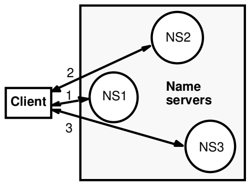
Flat name spaces

- Single global context and naming authority for all names
  - computer serial number
  - Ethernet address
  - remote object reference (IP address, port, time, object number, interface id)
- Names not meaningful
  - difficult to resolve (no tree hierarchy)
  - easy to create
  - easy to ensure uniqueness (timestamps)

Name Resolution

- Iteratively present name to a naming context,
  - start with initial naming context
  - repeat as long as contexts+derived names are returned
  - aliases introduce cycles (abandon after threshold no of resolutions or ensure no cycles)
- Replication
  - used for improved fault-tolerance on large services (more than one server, cf DNS)
  - may need navigation, i.e. accessing several servers

Iterative navigation

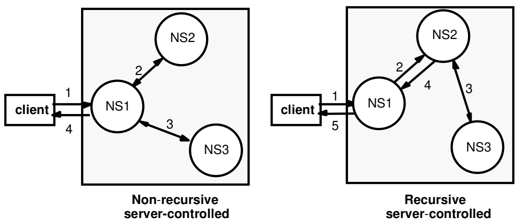


Database partitioned into servers according to its domain.
A client iteratively contacts name servers NS1–NS3 in order to resolve a name.
Servers returns attributes if it knows name, otherwise suggests another server.

Navigation methods

- Multicast navigation
  - client multicasts name to be resolved
  - server who knows name responds with attributes
  - problem: what if name unbound?
- Non-recursive server controlled
  - any name server can be chosen by the client
  - chosen server multicast/iteratively calls other peer servers
- Recursive server controlled
  - each iteration through a single server
  - calls continue recursively until resolution

Server controlled navigation



A name server NS1 communicates with other name servers on behalf of a client.

Replication & Caching

Replicate some directories for performance & availability.

- Updates
  - write to single master, master propagates updates
  - write to any replica: later merge updates (timestamps)
  - weak consistency (some entries out of date)
- Look-ups
  - try any local server: go to root and then down the tree
- Caching
  - names & addresses of recently used objects

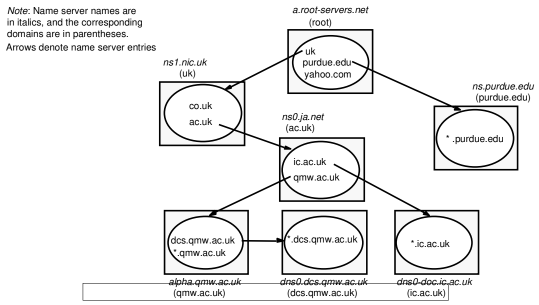
Internet Domain Name Service (DNS)

- Used mainly for host names and email addresses
- Extensible number of fields, separated by dot
  - gromit.cs.bham.ac.uk
- Host name resolution
  - resolves host name into IP address
- Mail host location
  - to resolve xxx@cs.bham.ac.uk, query DNS with domain name cs.bham.ac.uk and type ‘mail’
  - returns list of mail hosts, marked with preference value
- Reverse look-up (IP address to domain name)

DNS name servers

- Resource record holds
  - domain name for which record applies
  - time to live: initial validity time for cached entries
  - type (IP address, mail server, name server, alias)
  - value fields
- Replicated and partitioned information
  - update master server
  - Secondary servers
    - periodically download from master and save in cache
    - hold addresses of one or more masters up the tree
    - recursive look-up

DNS name servers



11 February, 2002 19

DNS summary

- DNS
  - relatively short average response time for look-ups
  - limited variety of data
  - infrequent changes in system
  - inconsistency of data possible (stale data may continue to be used)
- Problems (resolved in GNS)
  - rigid structure of the name space
  - lack of customisation of name space to local needs

11 February, 2002 20

Directory and discovery services

- Directory service
  - stores collections of bindings between names and attributes
  - provides look-up according to attributes (match all)
  - examples
    - Microsoft Active Directory Services X.500
- Discovery service
  - directory service that registers the services in a spontaneous networking environment
  - clients & services change dynamically
  - example: Jini discovery

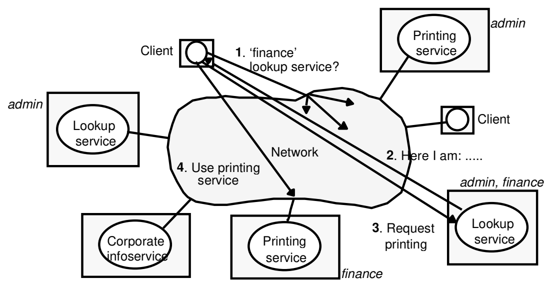
11 February, 2002 21

Jini discovery service

- Function
  - to enable users to access services (printing etc) from laptops while away, without their involvement
  - laptops look-up the services
  - services tell system of their existence and attributes
- Components
  - lookup service (registers and stores info about services)
  - Jini services (provide objects+attributes for the service)
  - Jini clients (request services that match requirements)
- Java/JVM based,
  - uses RMI plus download code

11 February, 2002 22

Service discovery in Jini



11 February, 2002 23

Jini

- How it works
  - services and clients join Jini dynamically
  - services have leases, which they have to renew periodically every *t* time units
  - look-up registers services (e.g. printer, what type, etc)
  - on entering, clients/services send request to multicast address
  - look-up services listen to such requests and reply with unicast address of service (e.g. printer)
  - client then contacts the service directly via RMI

11 February, 2002 24

Summary

- Name services
  - store names+attributes of objects, provide look-up
- Requirements
  - handle very large name spaces, long lifetime
  - high availability, fault tolerance
- Design issues
  - structure of the name space (syntax, resolution rules, is it changing over time)
  - distribution across servers, navigation
  - replication & caching

11 February, 2002 25