04 – Naming

- What’s in a name
- Naming System
- Design Tradeoffs

- Coulouris, Ch 9
- Craft, 1923
- Ahmed, 2005
- Subharthi, 2009

Naming

- Uses/affects
  - Identification
  - Location
  - Sharing
- Concerns
  - The name contains information?
  - Are names and references decoupled?
  - Can the system evolve?
  - Length
  - Syntax

Naming relevance

- What (and how) you can name
  - Fundamental to what systems do
  - Fundamental to how systems work
  - Fundamental to how systems evolve
- Uniformity
  - Naming and operations
  - Allow the operation of unknown things

Phone numbers

- Before 1999
  - Dynamic length
  - Local numbers
  - Prefix (outside calls)
  - xxx xxxx (within Lisbon)
  - 01 xxx xxxx (from the rest of Portugal)
  - +351 1 xxx xxxx (outside Portugal)
  - 0 1 YYYYYYYY (US)
  - 0 936 xxx xxx x

Phone numbers

- After 1999
  - fixed length (9 digits)
  - global numbers
  - 21 xxx xxxx (within Lisbon)
  - 21 xxx xxxx (from the rest of Portugal)
  - +351 21 xxx xxxx (outside Portugal)
  - 96 xxx xxxx

Phone numbers

- Differences
  - 310347
  - 21 3100347
  - 0936 990112
  - 96 9900112
- What's in the structure?
  - Local/global
  - Open/close
  - Reflects system?
- What can be accessed?
- Who generates names?
- What information is in the name?
- Are names locators?
  - Addresses?

- The nature of phone numbers determines
  - who can be called, and how easily
- Phone numbers embedded lots of information
  - Country where phone is hosted
  - Well known numbers have special use (911)
- Assignment is delegated to countries, etc
- The structure of a phone number helps with call routing
- The phone system has evolved in ways unimagined, and phone numbers mostly work

Naming

- name in a distributed system
  - string of bits or characters that refers to an entity
- Entities
  - Hosts, printers, disks, files,
  - Processes, users, mailboxes, news groups,
  - Web pages, graphical windows, message,
  - network connections,

Names

- Identifier
  - An identifier refers to at most one entity
  - Each entity is referred by at least one identifier
  - An identifier always refers to the same entity
- Address
  - The name of an access point to an entity
- Human friendly name
  - Unix file name, DNS names
- Names are always organized in a name space
  - A name space is an organization mechanism for a group of names

Names characteristics

- Uniformity (naming/access)
- Global vs. local names
- Absolute and Relative Names
- Opacity of names (purity)
- Name resolution

Uniformity

- Structure of a name should be constant
  - In space
  - In time
- Operations should also be uniform
- Allows extensibility and openness
- Phone numbers
- DNS
- URI/URN

Global vs. local names

- Global
  - Resolve to the same value everywhere
  - Absolute name
  - Required in consistent global systems
  - Location-independent naming
- Local
  - resolution depends on context
  - Relative name
- Phone number
  - Pre-1999 – Local
  - Pos-1999 - Global

Purity / Opacity of names

- Pure /opaque names
  - The resolution algorithm does not use any information on the name
  - Don't include any clue about the address/entity
  - Allow migration of entities
    - Resolution more complex
- Impure /tranparent names
  - Inofrmation on the name is used in the resolution
  - Name implies address/entity

Pure/Opaque names

- No special cases to avoid when allocating names
  - Any name usable for any object
- All names treated uniformly
  - often more efficient
- Separation of concern
  - the name solves one problem only…identifying an object

Impure / Transparent names

- It’s very handy to know something about an object from its name
- Names can reflect the structure of the system:
  - /comp/150IDS/files suggests a hierarchy of files for Comp Sci courses
- If consistent patterns are used, we can often predict one name from another
  - If http://weather.example.com/?city=miami is the Miami weather
  - …then maybe http://weather.example.com/?city=boston will get you Boston?
- Lots of different types of information winds up in names (for better or worse)
  - File type, parent directories, DNS name of host, operations
- But!
  - Only trust what the specifications say counts
    - In a URI, .html does not necessarily mean HTML, .jpeg need not be image/jpeg, etc.
    - mailto: definitely identifies a mailbox, because the specifications say it does in all cases

Purity / Opacity of names

- License plates (portugal)
  - Pure
- Phone numbers
  - Impure
- Ethernet
  - Pure
- URL
  - impure

Names / Addresses

- Name
  - an identifier for something
  - Not all names are addresses.
    - GUIDs, URNs, your name
- Address
  - a name that helps you locate something
    - HTTP URIs, “phone numbers”, postal addresses
- For names that are not addresses:
  - resolving the name to an objectis necessary
    - hash table, registry, associative search, etc.

Collision / Alliasing

- Collisions
  - using one name for multiple things
- Aliasing
  - Using several names for a single thing

Aliasing

- Can be handy
- Even for simple reasons: http://example.com vs HTTP://EXAMPLE.COM
- Absolute vs. relative names for the same thing
- a/b/c vs a/b/../b/c
- Creating more than one name for an object tends to cause confusion
- Reasoning about systems with aliases is much harder
- Even allowing for multiple names causes trouble
- If I have a cached copy of http://example.com, is it usable for references to HTTP://EXAMPLE.COM? -- cache code must be prepared to check
- Computer languages that allow pointer aliasing are very hard to optimize… it's very hard for the compiler to keep track of which pointers might be referencing a given part of memory
- On the Web: aliases make it hard to tell which pages are popular

Collision

- Should be allowed?
  - Usually not (no unrelated objects)
- When is it useful?
  - Multiple versions of a page that changes
  - The mobile vs. full size version of a Web page
  - A resource (a movie) vs information about the resource (blog about the movie)
- But
  - we lose the ability to reliably pick out one or the other

Should all addresses resolve

- Depends on the system
  - Some programming languages will create a variable when you access it
  - Some hypertext systems won’t let you make a link unless the target exists
    - Indeed, Tim Berners-Lee’s early work on the Web was rejected by academics because he didn’t insist on such consistency…

Dangling links and “404” status code

- One of Tim’s key contributions was the realization that
  - enforcing such consistency is impractical on a global scale
- You can have a global Web, or you can have all links resolve…
  - but not both!
- That’s why the 404 HTTP failure code is one of the most important architectural features of the Web!

Name resolution

- The process of looking up a name
- Iterative
  - Start with the root
  - Each layer resolves as much as it can and returns address of next name server
  - Client interactively contacts servers
  - Lower performance demand on each name server
- Recursive
  - Start at the root
  - Each layer resolves as much as it can and hands the rest to the next server
  - Servers recursively contacts servers
  - Caching possible at name servers
  - Communication cost reduced

Iterative name resolution

- Start with the root
- Each layer resolves as much as it can and returns address of next name server
- Iteractively client concats servers

Why phone numbers?

- Good example
- Much like the web
  - Scale: worldwide
  - Time horizon: centuries
  - Evolving:
    - New devices, new operators
  - Federated
    - Distributed organization (countries, telcos)

DNS

- Primarily used for looking up host addresses and mail servers
- Comparable to a telephone book
- Its name space is hierarchically organized as a rooted tree
  - Each zone is implemented by a name server, which is always replicated
- DNS also maintains an inverse mapping of IP addresses to host names