What is an information object anyway?

4WARD WP6
Network of Information

Börje Ohlman
(Ericsson Research)

Bengt Ahlgren
(Swedish Institute of Computer Science)

& 4WARD WP6 colleagues
- Combination of clean-slate research approaches to address the **Network of the Future**
- **Size:** Roughly 23 M€
- **Time frame:** 2008-2009
What is an information object anyway?

- What are the primary components of an information centric network?
- How should an information object be named?
- Desirable properties of a name
- Components of a proposed naming scheme
Identifiers and Information Modeling

- **Persistently identify information**
  - Location-independent identifiers
  - Represent *multiple copies*

- **Representation of information via Information Objects (IOs)**
  - Another level of indirection
  - Represent information **independent** of a *specific copy*
    - *E.g. a text, a song*
  - Consists of a set of attributes
    - Including media components, GPS location, access rights, encoding

- Information Objects can also represent:
  - Streams
  - Services
  - Real-world objects (e.g., a physical copy of a book, a person)

- IOs can be used to organize information
Organize Information – IO, DO and BO definitions

Information Object (IO)
- An Information Object is a set of attributes defining the semantics of a data object. An IO may refer to a piece of music, a film or a webpage.
- Can be static, dynamic or real-world objects, including streams and services.

Data Object (DO)
- Sub-class of IO holding attributes for bit-level objects and pointer(s) to the actual data.

Bit-level Object (BO)
- A specific sequence of bits, independent of any semantic meaning, also independent of where they exist, like in a file, on the wire, in the air or in a primary memory.
Naming Requirements

- NetInf prioritized:
  - Self-certification and self-generation
    - Reduce the need for trust in the infrastructure
    - Data integrity
    - No need for a new naming authority
  - Persistent names, inert to:
    - Owner change
    - Content change
    - Algorithm changes (hash or crypto)
  - Support for all types of information objects:
    - Real world, Services, Streams, Static files, Dynamic data
  - Globally unique names

- More on the wish list:
  - Owner/Publisher authentication
  - Variable length
  - Human readable
  - Minimize load on (Name Resolution) infrastructure
Naming tradeoffs

- Self certification vs. Dynamic objects, Real world object, Services
- Persistent naming vs. Self certification & Dynamic objects
- Verifying ownership by naming vs. persistent names and allowing change of ownership
- Simplicity vs. flexibility
NetInf Naming

| Tag | P=Hash(PublicKey_{Owner}) | L={Hash(C) | String} |

Tag
- Defines the format
  - Hash algorithm used (SHA1, MD5, ...)

Principal (P)
- Object ‘publisher’ (optional)
  - Owner
  - Creator
  - Anonymizing service

Label (L)
- Identifying individual object published by Principal
  - Hash of object or label created by principal

IDs have no hierarchical structure
Strong influence on name resolution!
Two approaches to name resolution

- **Multiple DHTs (MDHT)**
  - Hierarchical DHTs (Provider-based)
  - Topological embedding of DHTs
  - Name-based routing

- **Late Locator Construction (LLC)**
  - Attachment registers to keep track of immediately attached neighbours
  - Hierarchical locators constructed on demand at the time of session initiation
Conclusion

- Design of a new network architecture based on information-centric paradigm
  - Rather than based on a host-centric paradigm

- Some characteristics of Networking of Information (NetInf)
  - Information model: information object, data object, bit-level object
  - Naming scheme for naming information objects independent of nodes
  - Scalable solution for node and network mobility and multihoming
  - Enable efficient information dissemination
    - Benefit from available copies, anycast, solve Flash-Crowd Effect, …
  - Secure information-centric architecture by embedding security into identifiers
  - A common infrastructure and API for accessing all types of objects (including real world objects), regardless of their location
  - Scalable name to locator resolution for $10^{15}$ objects and beyond
  - Designing NetInf to make it largely self-managing
Thank you for your attention
Backup Slides
NetInf combined name resolution and attribute search

- **IO** = Information Object
- **DO** = Data Object
- **BO** = Bit-level Object
WP6 – NetInf publications

- **Providing Data Dissemination Services in the Future**, M. D’Ambrosio, P. Pasano, M. Marchisio, V. Vercellone, M. Ullio, WTC'08
- **Design Considerations for a Network of Information**, Bengt Ahlgren, M. D’Ambrosio, C. Dannewitz, M. Marchisio, I. Marsh, B. Ohlman, K. Pentikousis, R. Rembarz, O. Strandberg, V. Vercellone, ReArch '08
- **Distributed Information Object Resolution**, K. Pentikousis, ICN 2009
- **Private Domains in Networks of Information**, R. Rembarz, D. Catrein and J. Sachs, Future-Net'09
- **Self-management for a Network of Information**, K. Pentikousis, C. Meirosu, A. Miron, and M. Brunner, Future-Net'09
- **Energy-efficient Multiaccess Dissemination Networks**, K. Pentikousis, GreenComm'09
API for accessing any type of object, regardless of location

NetInf

Internet  Mobile networks  Broadcast TV/Radio  Real world

ggetObject(objectID)
ggetObject(attr1, attr2, attr3)
NetInf Architecture Overview

NetInf node

Applications

Application programming interface
publish (...) resolve(...) join (...)

NetInf API

NetInf Storage API
Storage engine

NetInf App X API
App X engine

Resolution engine

Local storage engine

Local resolution engine

Transport control engine

Cache engine

TCP/UDP/IP
IP multicast
Generic Path(s)

NetInf Additional Services
Storage protocol(s) STORE(...) NetInf App. X protocol

Name resolution protocol(s)
PUT(...) GET(...) ...

NetInf Transport control protocol

NetInf Generic Path Factory
World-wide Scalable Name Resolution using Multiple DHTs (MDHT)

- Combination of:
  - Hierarchical DHTs (Provider-based)
  - Topological embedding of DHTs
  - Name-based routing

[Diagram showing a Name Resolution System with nodes HOST A, HOST B, Local Resolution, Global Resolution, Source Region, and Destination Region. The diagram illustrates the process of getting data (Get(X)) and returning data (Return data).]
## DONA vs. MDHT

### Performance and Scalability

<table>
<thead>
<tr>
<th></th>
<th>DONA</th>
<th>MDHT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registered Items</td>
<td>$10^{11}$</td>
<td>$10^{15}$</td>
</tr>
</tbody>
</table>
| Storage Memory per Node | 4 TB at Tier 1  
<4 TB at lower Tiers | 4 TB on all network nodes of the global Internet |
| GET Rate         | 20K requests per Gbit/sec.  
i.e. 2 Requests/sec per user?? | 2 Requests/sec per user with current storage technology  
(better results with parallelization) |
| Number of users per node | No information published | O($10^4$) users per node with a rate of 2 Requests per second per user and current storage technology  
(better results with parallelization) |
| Refresh Process Bandwidth | No information published | 10 Mbps                                  |
| Refresh TTL      | No information published | <6 days                                   |
Mobility network state
LLC vs. DONA

LLC

DONA

NetArch’09 15-20 March 2009
Mobility network state
LLC vs. DONA

LLC

Tier 1 providers

AR

RH

RH

RH

Host 1

Host 2

Host 3

Data Object Zn

Data Object Yn

Data Object X

DONA

Tier 1 providers

RH

RH

RH

Host 1

Host 2

Host 3

Data Object Zn

Data Object Yn

Data Object X
Problems Resulting from a Host-centric View

- No common *persistent naming scheme* for information
  - Information is named relative to the box they are located in, URLs resolves to IP-addresses
    - Moving information = changing its name ("404 file not found" errors)
- Mobility and multihoming for hosts and networks is problematic due to the semantic overload of IP-addresses
- No consistent *representation of information* (copy-independent)
  - No consistent way to keep track of *identical copies*
  - Different *encodings* (e.g., mp3, wav) worsen problem
- Security is host-centric
  - Mainly based on *securing channels* (encryption) and *trusting servers* (authentication)
  - Can’t generally trust a copy received from an untrusted server

Problems can be solved in a consistent manner via an information-centric architecture
Future work

- Finalise security evaluation of naming scheme
- Define a service model building on the object model
  - Showing how services are embedded into the NetInf architecture
- Search?
- Extensions for delay-sensitive applications
  - Purpose: see how far into real-time domain the NetInf concept can reach
- Routing approaches:
  - Finalising designs, defining how they are combined into the same system design
  - Perform extensive evaluation, primarily with simulation, but some implementation
- Overall performance evaluation
  - Simulation of personal mobile scenario and cooperative multi-access
  - Finalise simulation setup and produce results (T6.5)
  - Defining metrics and exploring parameter space
- Proof-of-concept prototyping
  - Serverless web and personal mobile scenario
  - Small scale performance evaluation, corroborating and providing parameters to the simulation
Organize Information – Examples of IOs and IO Hierarchies

- Generation of IOs and IO hierarchies:
  - Content owners, community-based (see e.g. Wikipedia)
Versioning and Revocation

Deletion challenges:
• Multiple copies
• Disconnected operation
• Central register?

Possible strategies:
• Self-deleting objects
• Objects needing recertification
• Invalidation of decryption key