

# Service/Resource Naming a Comparative Study

Reaz Ahmed, Raouf Boutaba, Fernando Cuervo,  
Youssef Iraqi, Dennis Tianshu Li, Noura Limam, Jin  
Xiao & Joanna Ziembicki

School of Computer Science  
University of Waterloo



1

University of Waterloo, Waterloo

## Abstract

This presentation will outline our groundwork in designing a naming scheme for a cross-domain, Internet-scale service discovery framework. We will present a critical look at a sample of existing approaches in naming of services and resources and we will compare them against each other based on what we identify as relevant criteria. Finally, we will present our choices for suitable candidates.

## Keywords

Service and Resource Naming, Name Resolution Architectures, Naming Authority

## Authors

Reaz Ahmed	r5ahmed@uwaterloo.ca
Raouf Boutaba	rboutaba@bbcr.uwaterloo.ca
Fernando Cuervo	fernando.cuervo@alcatel.com
Youssef Iraqi	iraqi@bbcr.uwaterloo.ca
Dennis Tianshu Li	dtianshu@bbcr.uwaterloo.ca
Noura Limam	nlimam@bbcr.uwaterloo.ca
Jin Xiao	j2xiao@uwaterloo.ca
Joanna Ziembicki	jziembi@uwaterloo.ca

Acknowledgements: This work has been sponsored by Alcatel Inc. and the Natural Sciences and Engineering Research Council (NSERC) of Canada.

# Objective

*To devise a scheme for naming resources and services in an Internet-scale multi-domain environment*

- *Service*
  - A set of functionalities associated with a process or system that performs a task.
- *Resource*
  - An entity that is used or acted upon by a process or system.
- *Domain*
  - A collection of resources and services controlled and managed by a single administrative entity.

2

University of Waterloo, Waterloo

The following presentation will examine existing naming approaches, with a brief summary of each approach. Subsequently, we will propose a set of comparison criteria that work well with our objectives, and compare the approaches against these criteria. Finally, these criteria will be used to select a set of approaches that we believe are the best match with our objectives. We will see that no single approach currently deployed on the Internet perfectly matches our objectives, but that some approaches can be combined with others, and extended in order to meet our goals.

Our goal is to search for the best possible way to name services and resources in multi-domain, Internet-scale applications. Current URL scheme is unsuited for next generation services with its location dependence, domain-specific name assignment, etc. At the same time DNS is being overtaxed with increasing request volume. Its hierarchical resolution architecture does not scale at the root DNS level. With the additional characteristics of next generation services such as mobility, context-awareness, etc. new naming schemes and name resolution architectures are needed.

In order to name these entities, it will be helpful to first define what we mean by services or resources. We look at a service as a set of functionalities provided on behalf of the user. This set of functionalities could involve, for example, a service provided by a human (like phone technical support, or haircutting), but in our context, a service would normally be implemented by a software application.

Now, a resource can be looked at as an input that is used by a service to produce an output. A resource could be a physical entity such as a fibreoptic link or hard drive storage, or a logical entity such as a routing table or employee data in a database.

# Terminology

- ❑ **Name**
  - A linguistic object that singles out a particular entity from among a collection of entities
- ❑ **Address**
  - An identifier that allows an entity to be reached
- ❑ **Namespace (address space)**
  - The collection of all valid names (addresses).
- ❑ **Name Resolution**
  - The mapping between a namespace and an address space.

3

University of Waterloo, Waterloo

In our definition of a name we focus on its uniqueness, which means that a name can refer to only one entity (or only one collection of entities, where this collection is treated as a unit). A name could also denote an equivalence class of objects.

As we will see, a “name” can refer to a number of different concepts: it can be a simple identifier (Kate), include description (the girl who works in the Networks lab, hair=brown, eyes=brown, height=160cm), or include location information (Kate, office =DC3335B, phone number=888-4567).

An address, on the other hand, is a specific form of a name that provides information on how the entity can be reached. In general, an address is the identifier that is one step closer than a name in the process of reaching this entity; for example, if a URL is a name, then the IP address combined with a path is considered as an address. On the other hand, if we consider an IP as the name of a network interface, the MAC address would be considered an address, because it takes us one step closer to physically reaching that interface.

The namespace is the collection of all valid names, and the address space is the collection of all valid addresses.

One of the most important issues, which we will examine in detail during this presentation, is name resolution. Name resolution is the act of translating a name into its corresponding address; it also necessarily concerns which names map to which addresses.

# Naming Approaches (1)

- ❑ Universal Unique Identifier (UUID)
  - 128-bit number, locally-generated, guaranteed global uniqueness until 3400 A.D.  
`58f202ac-22cf-11d1-b12d-002035b29092`
- ❑ Uniform Resource Identifier (URI)
  - URLs (Uniform Resource Locators) -- location-dependent
    - `<scheme>://<user>:<password>@<host>:<port>/<url-path>` (for IP-based protocols)  
`http://www.w3c.org/Addressing/`
  - URNs (Uniform Resource Names) -- location-independent
    - `<URN> ::= "urn:<NID>:<namespace specific string>"`  
`urn:isbn:0-395-36341-1`

4

University of Waterloo, Waterloo

In our comparison, we will discuss the following naming approaches: the Universal Unique Identifier, the Uniform Resource Identifier family (which includes URLs and URNs), the XORP Resource Locators, which are used in an open router software project, and two descriptive naming schemes: INS and Solar, that do not precisely match our definition of a naming scheme, but are useful to include for comparison. The first considered naming approach is UUID, which was first used in Network Computing System (1987).

1. The UUID [1] is a 128-bit flat identifier that is generated independently for each object. This means that each host or software component can generate its own unique identifier, according to a number of algorithms that involve the timestamp, the MAC address and a random component. The algorithms are claimed to guarantee uniqueness of UUIDs across time and space without requiring a central registration process.
  - One of the most prominent applications of the UUID is in Web Services, where it uniquely identifies each service.

2. The URI family is the most often used naming system in the Internet, and dates back to the beginnings of the World Wide Web. It is defined by the IETF URI working group formed in 1994. It includes a number of sub-families such as URL and URN.

The Uniform Resource Locator [2] is widely deployed and adopted by the W3C. It consists of the Internet application protocol required to access the resource (http, ftp, etc.), the domain name of the host that provides the resource and the pathname of the resource on that host.

The Uniform Resource Name [3] provides persistent, location-independent identification. It consists of the string "urn:", the Namespace Identifier (NID) and the Namespace Specific String (NSS). Typically, Namespace Identifiers are registered with and assigned by the IANA [9], and identify the category of the resource. The Namespace Specific String uniquely identifies the named service or resource.

## Naming Approaches (2)

### □ Grid Service Naming

- Grid Services use two types of URIs:
  - GSH (Grid Service Handle)
    - Location-independent global names
  - GSR (Grid Service Reference)
    - Location-dependent access points (may also take form of WSDL documents)

### □ XORP Resource Locators (XRL)

- XRLs are used for inter-process communication
  - Unresolved form: `finder://fea/fti/0.1/add route? net:ipv4net=10.0.0.1&gateway:ipv4=192.150.187.1`
  - Resolved form: `xudp://192.150.1.5:1992/fti/0.1/add route? net:ipv4net=10.0.0.1&gateway:ipv4=192.150.187.1`

5

University of Waterloo, Waterloo

3. The Web Service architecture specification, formalized in Jan. 2002 by the W3C, defines a web service as a “software system designed to support interoperable machine-to-machine interaction over the network”. Web Services communicate over HTTP using XML documents and perform tasks on behalf of the human users. Grid Services [4] are a specific form of Web Services, and are designed for large, but fairly tightly-bound distributed computing environments. They support stateful services, allowing explicit creation and destruction of service instances.

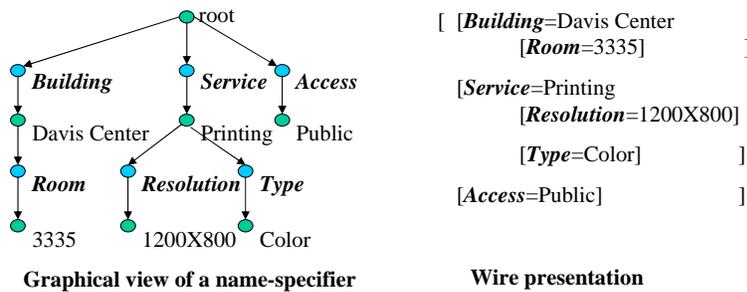
Although Grid Services use the previously described URI naming scheme, we have decided to mention them separately because of their importance as service-providing mechanism. We present here the two-layer Grid service naming scheme, whose components -- GSH and GSR -- are described below.

- The Grid Service Handle (GSH) is a globally-unique name that distinguishes a specific Grid service instance from all other Grid service instances that have existed, exist now, or will exist in the future. A GSH is often viewed as a permanent network pointer to a particular Grid service instance. A GSH is a minimal name in the form of a URI and may not carry enough information to allow a client to communicate directly with the service instance.
- The Grid Service Reference (GSR) contains all information that a client requires to communicate with the service instance via one or more network client-specific protocol bindings. Before using a Grid service, a GSH must be resolved into a GSR.

4. XORP (eXtensible Open Routing Platform) is an experimental open-source software router platform (v. 1.0 released in 2004). XRLs (XORP Resource Locators) [5] are used to mediate IPC (Inter-process Communication) within XORP processes. XRLs are similar in structure to URLs. Instead of representing a hierarchical resource location, however, an XRL uses a specific syntax to describe an inter-process procedure call and its parameters. An XRL contains the protocol family to be used for transport, the arguments for the protocol family, the interface of the target being called, the target's version, the method, and the argument list.

## Naming Approaches (3)

### □ Intentional Naming System (INS)



### □ Solar

- Names may include variable portions, for example:

[Device=Laptop, Owner=Bob, location=\$building:\$room]

Can be resolved to

[Device=Laptop, Owner=Bob, location=Davis Centre:3335]

6

University of Waterloo, Waterloo

5. The Intentional Naming System [8] was developed at MIT in 1999. The goal of INS is to describe *what* the user is looking for, not *where* to find it. It is a descriptive naming scheme, where each name is a hierarchical arrangement of attribute-value pairs.

- The naming scheme aims specifically to be location independent; so much so that binding to location can be deferred to when a message is actually being delivered.
- Wild card token (\*) and inequality operators (<, > etc.) are allowed in name-specifiers to form query expressions.
- The INS/Twine extension to the resolution mechanism is intended for dynamic networks on the order of hundreds to thousands of nodes.

The INS scheme, as it stands, is not necessarily a naming scheme as we define it, because it does not enforce uniqueness of a name -- it is more of a resource discovery scheme, where all the resources satisfying a user's request have the same name. However, this is up to implementation. The system can be implemented with intentional anycast, the name resolution system selects a single "optimal" entity, based on some selection metric, which ensures a sort of name uniqueness.

6. Like INS, Solar [6] is an attribute-value descriptive naming scheme. It is built on top of INS and adds the capability for a name to include variables that change over time. The variable parts of a name are used to provide context information, typically the location of a mobile resource or service. These variables are evaluated each time the name is referenced.

## Criteria for comparison

- ❑ Readability
- ❑ Extensibility
- ❑ Namespace size
- ❑ Naming authority
- ❑ Name resolution
- ❑ Name persistence
- ❑ Standardization and acceptance

7

University of Waterloo, Waterloo

Based on our study of naming schemes and the requirements of our target application, we have identified a set of relevant comparison criteria. We will use these criteria to compare, contrast and analyze the presented naming approaches, in order to gauge their suitability for a large-scale multi-domain naming scheme.

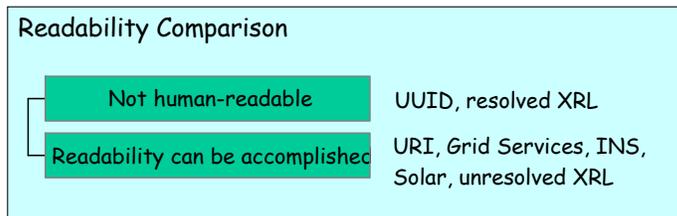
The comparison criteria we have identified are, briefly:

- Readability: are the names human-readable? Are they human-friendly?
- Extensibility: Can the format or structure of names be updated while remaining compatible with old names?
- Namespace size: how many objects/entities can we uniquely name?
- Naming authority: who assigns the names?
- Name resolution: how are names mapped to addresses?
- Name persistence: do the names permanently denote the same entity?
- Standardization and acceptance: how well is the naming scheme defined and accepted in the Internet community?

After examining naming schemes based on these criteria, we will use the results of the comparison to propose a solution that would best fit the needs of service naming in a large-scale multi-domain environment.

# Readability

- ❑ Are the names human readable or not human readable? If not, are they human-transcribable?
- ❑ How much helpful information (routing, descriptive) is contained in the name?
- ❑ All schemes are human-transcribable, and except for UUID (and some forms of XRL) they can accomplish readability



8

University of Waterloo, Waterloo

Names may or may not be considered human-readable, depending on whether they provide useful information to the humans who use them, and whether they are memorable and comprehensible to humans. Readability is determined by the set of allowable characters (alphabet) in a naming scheme, as well as the format and structure of a name.

A somewhat weaker requirement is for a name to be human-transcribable -- it does not have to be easy for a human to remember such a name, but it must be possible for a human to, for example, take a pen and copy the name out on a piece of paper.

Moreover, a naming scheme may define names that are strictly human readable, strictly human-unreadable, or it may allow both kinds of names. The readability/human-friendliness of an individual name is a subjective measure, influenced by the authority assigning the name, however, some naming schemes may lend themselves better than others to direct human use.

For example, the URL

<http://www.example.org/58f202ac-22cf-11d1-b12d-002035b29092.txt>

is not human-readable, but

<http://www.ietf.org>

is human-readable

In URIs, non-ascii characters can be escaped by %, so even if a name is not human-readable, it is human-transcribable.

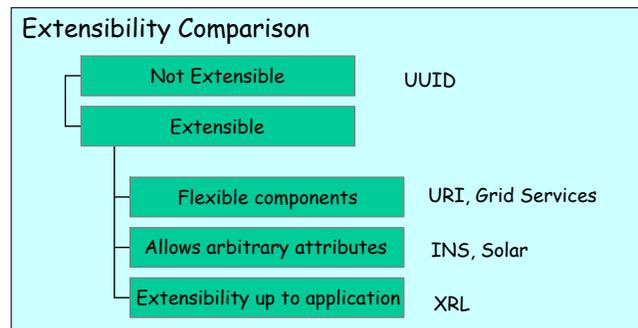
In XRLs, the unresolved form is considered human-readable, but the resolved form is not considered human-readable (or human-friendly), because it may contain network-specific addresses.

In INS, names can be expressed in several different ways: as a tree, in attribute-value bracket notation, or in XML.

In Solar, meaningful variable names can improve the readability of the variable portions of the name

# Extensibility

- ❑ Can the format or structure of the names be updated, while remaining compatible with old names?
- ❑ Can the scope be extended (naming new types of services and resources)?



9

University of Waterloo, Waterloo

The naming system may be required to be extensible for future updates. Updates can occur in the format and structure (e.g. more bits are allocated to the name), in the scope (i.e. introduction of new types of services or resources to be named) and in the grammar generating the namespace.

In a quickly-growing and quickly-changing environment like the Internet, it is desirable for naming schemes to be extensible, so that new names remain compatible with existing, older names from the same naming scheme.

The UUID contains no descriptive information, and can hence be used to name a variety of resources and services. However, due to its finite namespace size, as discussed in the next section, it is difficult to extend once the namespace is exhausted.

•On the other hand, some schemes, such as URI, are quite extensible, and standardization efforts exist for their extensions. Proposed URI extensions include:

-Use of URIs as identifiers for non-network resources (for example, to identify an abstract object such as an XML namespace, or a physical object such as a book or a person).

-IRIs (Internationalized Resource Identifiers): the extension of URI syntax to non-ASCII characters, in order to allow the use of languages which do not use a Latin alphabet.

•INS and Solar are extensible because they do not assume a predefined set of attributes.

•XRL's extensibility depends on the XORP API

## Namespace size

- ❑ A set maximum name length implies a finite namespace
- ❑ All mentioned naming schemes have infinite namespaces, except for UUID (128 bits, hence  $2^{128}$  possible names)



10

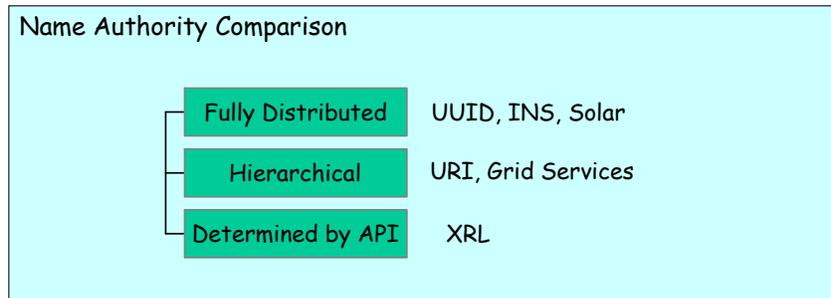
University of Waterloo, Waterloo

The size of the namespace is related to extensibility. It determines how many entities can be uniquely named. A limit on the size of a name implies that the namespace is finite. A finite name size allows devices to allocate a fixed amount of space for storing and transmitting names, which can improve performance and reduce implementation complexity.

However, a finite namespace runs the risk of being exhausted, as evidenced by the impending depletion of the IPv4 address space. In a multi-domain environment, a naming scheme should have a namespace that is large enough to uniquely denote all existing and future entities in its desired scope.

## Naming authority

- ❑ Can be centralized, hierarchical, or fully distributed
- ❑ In a fully distributed scheme, names are assigned independently from each other; each entity can be its own naming authority



11

University of Waterloo, Waterloo

A naming authority is the entity that assigns and manages names in a namespace. It may be centralized (a central authority assigns and manages all the names), hierarchical (each domain is responsible for naming the resources and services under its own management), or fully distributed (the name for each entity is chosen independently, or even, each entity chooses a name for itself). Regardless of its management scope, a naming authority is responsible for assigning and managing names for resources and services.

Distributed name authority may or may not ensure uniqueness. The generation algorithm suggested for UUID is claimed to support very high allocation rates of up to 10 million requests per second per machine if necessary, until AD3400 [1]. On the other hand, INS and Solar make no claims towards name uniqueness.

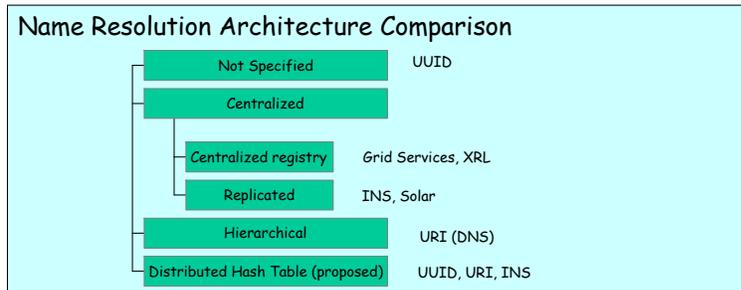
For use in Internet-wide applications, URL scheme identifiers such as *http* or *ftp* are registered with IANA. Furthermore, the Domain Name portion of a URL must be registered with the Internet Corporation for Assigned Names and Numbers (ICANN). Finally, the remaining portion of a URL is assigned hierarchically within a domain.

An OGSA (Open Grid Service Architecture) GSH is a valid URI, hence it follows the hierarchical approach used in URI name assignment. Also, since a factory approach is used to create a transient service instance for each client, it is important to consider how to uniquely name every service instance. OGSA uses globally unique names (URLs) combined with locally-generated identifiers (usually a large hash value) to form a valid URI that can uniquely identify Grid service instances.

The protocol family part of an XRL is defined by the XORP API. In the runtime environment the finder process works as the naming authority for handling module names, interface names, interface version and method names.

# Name resolution

- Name resolution -- translation of names to addresses
  - Architecture: centralized or distributed?



- Scalability: Will the system handle growth?
- Efficiency: Can the system provide fast response and an acceptable level of performance?
- Fault-tolerance: Can the system retain most of its functionality after failure of some of its components?

12

University of Waterloo, Waterloo

Name resolution translates names to the addresses that allow a user to access the corresponding entities. It has two main aspects: *name-to-address mapping* and architecture.

Name to address mapping refers to the relationship between names and their corresponding addresses; it can be deterministic, such that a name always resolves to the same address, or nondeterministic, such that a name can resolve to one of several addresses pointing to the referred entity. Additionally, a name may be location-dependent or location-independent.

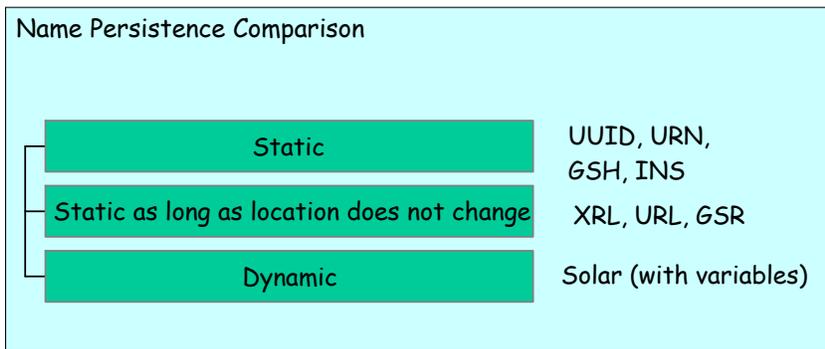
The implementation of name resolution involves the physical and logical architecture of the resolution system. This architecture may be centralized, hierarchical, entirely distributed (peer-to-peer) or a combination (e.g. partially-centralized peer-to-peer). The choice of name resolution architecture determines the system's scalability, efficiency, and fault-tolerance.

- The UUID proposal does not specify a name resolution architecture. However, a flat naming scheme such as UUID lends itself well to resolution using a Distributed Hash Table (DHT).
- URLs are resolved using DNS, although a new, DHT-based approach (CoDoNS) has recently been proposed [10].
- URNs can be resolved (e.g. into URLs), using a variety of approaches, including a Naming Authority Pointer in the DNS record.
- Grid Services use a logically centralized (but potentially, physically distributed) registry.
- In INS and Solar, resolution is done by lookup on the name tree in one or more Intentional Name Resolvers (INRs). In the case where there are multiple INRs, the entire name tree is replicated in each INR. The scheme is, hence, a replicated centralized registry. A DHT-based scheme, called INS/Twine, has also been proposed.
- XRLs are resolved within the XORP application using the IPC finder service.

Each type of name resolution architecture has its advantages and disadvantages. A centralized name resolution scheme is naturally consistent and easy to maintain, but does not scale well and introduces a single point of failure. A hierarchical name resolution scheme such as DNS has natural bottlenecks at the higher-level servers (especially the root servers), but the bottlenecks can be partially remedied by using replication (multiple replicas of the root server) and caching. Finally, a DHT-based scheme has fewer bottlenecks, but can be expensive to maintain and incur higher lookup costs than a hierarchical scheme.

## Name persistence

- A *static*, or persistent name permanently denotes the same entity
- A *dynamic* name is assigned for a limited period of time



13

University of Waterloo, Waterloo

Names can be divided into two categories: static names and dynamic names.

A *static* name is a name that permanently denotes the same entity. This is a very usual property of names and is often implied.

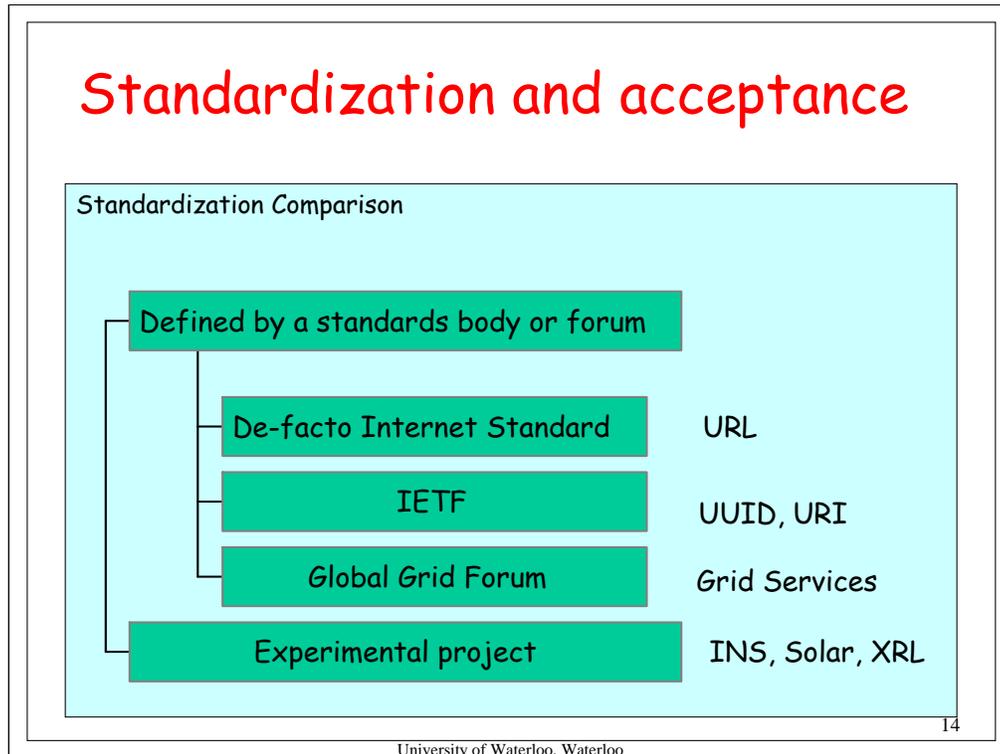
A *dynamic* name is a name that is assigned to an entity for only a limited period of time, which is short with respect to the lifetime of the entity.

For example, a given name is a static name, because it is assigned permanently at birth (barring, of course, legally changing one's name or marriage). If, on the other hand, one assigns consecutive numbers to people in a room based on where they are sitting, these are dynamic names. If a person gets up and switches to a different chair, then the name is no longer valid. Another example of a dynamic name is a web page whose filename changes every week (e.g. articles in an online newspaper).

Location-dependence usually implies that the name is up to a point, dynamic, but whether the name is static or dynamic may also depend on other factors. For example, a certain segment of memory may be given a variable name for the duration of the program execution, but outside the scope of the program, the name is no longer valid.

In systems that use dynamic names, an important issue is ensuring that the name resolution information changes when name mapping changes. This issue is similar to the problem of distributed garbage collection.

# Standardization and acceptance



A naming scheme that has a well-defined standard, and is easy to incorporate into the existing infrastructure is far easier to deploy than an experimental, constantly-changing scheme, even if the latter is superior in other aspects. Ideally, a multi-domain service and resource naming scheme would be accepted by an official standardization body, and would have most components of its name resolution already in place.

- UUID is defined by the IETF, and widely used. For example, the service discovery systems UPnP, Jini, and JXTA, and Bluetooth's SDP, as well as Web Services, all use UUID as service/resource identifiers.
- The URI specification [7] is being updated for advancement to IETF Standard. URL and URN are also defined in RFCs on the IETF standards track (RFC 2396, RFC2141) and is the primary scheme used for naming resources on the Internet. The Domain Name System used in URL schemes is an Internet Standard. URL scheme identifiers and URN namespaces are managed by IANA.
- GSH and GSR are standardized by the Global Grid Forum in the OGSI v1.0 Specification. It is worth noting that with the proposal of the Web Service Resource Framework (WSRF) it appears that Web Services and Grid Services are gradually converging.
- XRL is a part of XORP, an experimental project that has not yet been standardized. Version 1.0 of the XORP platform, the first version recommended for use in real-world routing, was released in July 2004.
- INS was developed by the MIT Laboratory for Computer Science in 1999 [8], and so far has not been standardized. INS provides an API for name-specifier manipulation and discovery process. Implementation and source code for INS can be obtained from the project website .
- Solar is a research project that aims to achieve context-sensitive resource discovery in a pervasive computing environment. Its specification was published in the First IEEE International Conference on Pervasive Computing and Communications, 2003 [6]. However, this naming scheme has not been standardized.

