Is It Possible to Use One Name for Many Sites?

L. Preechaveerakul
Department of Mathematics, Faculty of Science
Chulalongkorn University, Thailand 10330
Email: Ladda.Pr@Student.chula.ac.th
Phone: (662)2185144 Fax: (662)2552287

P. Bhattacharosol
Department of Mathematics, Faculty of Science
Chulalongkorn University, Thailand, 10330
Email: bpattara@sc.chula.ac.th
Phone: (662)2185144 Fax: (662)2552287

Abstract

Names play an important role in all computer systems to resolve the objects. A naming system is necessary to implement. Although the existing naming system are implemented both in distributed and non-distributed systems, most of their structure are hierarchical. The structure obtains 3 limitations and drawback: uniqueness, scalability, and anonymity. In real world, the organizations with many branches sometimes require the situation called one name – many objects – one result for its unity. However, most of the current naming systems do not serve it. Therefore, this paper proposes a new naming system called Sharable Name System that can be solved the limitations and drawback and maintains unity of the organizations. With this proposed solution, the extension of network systems will not be limited.

Keywords Naming System, distributed system

1. Introduction

Names have several purposes in any systems. An important purpose is to facilitate referring the objects. Names are used to share resources, to uniquely identify entities, to refer to locations, and so on [17]. In the computer world, a vast interconnected collection of host computers called Internet, needs names to refer to the addresses. An open protocol standards, TCP/IP protocols [6, 14] are comforted machines from anywhere to communicate. Standard and non-standard application protocols have been developed for services especially, name service which is a fundamental service to all computer networks. A variety of name service has been developed to map physical resources to logical names. Global Name Service (GNS) [7], Handle System [16], Network Information Service Plus (NIS+) [13], Novell Directory Service (NDS) [1, 8] and Domain Name System (DNS) [9, 10, 11] have hierarchical name space. We summarize the features of each name service in Table 1 and found their limitations.

We conclude limitations and drawback of them in section 2. We propose the new naming system called Sharable Name System (SNS) which address limitations and drawback of current name services in section 3. This paper contributes for the name service that can be shared:— one name to many objects. In section 4, we discuss our work. Finally, we give our conclusion.

2. Limitations and Drawback of Various Name Services

Since the structure of each name service referred in section 2 is hierarchical. Most of them have the limitations and drawback that we categorize into 3 groups: uniqueness, scalability and anonymity.

Uniqueness
The structure of each name service is hierarchical; therefore each name in a tree must be unique.

Scalability
Most of the existing naming system was designed for people from a single language (English), not other languages.

Anonymity
It is sometimes necessary to say things anonymously [5]. The name service structure is hierarchical so that it is easily traced back to its ancestors in the tree.

<table>
<thead>
<tr>
<th>Table 1 Features of the existing name service.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name Service</td>
</tr>
<tr>
<td>Global Name Service (GNS)</td>
</tr>
<tr>
<td>Handle System</td>
</tr>
<tr>
<td>Network Information Service Plus (NIS+)</td>
</tr>
<tr>
<td>Novell Directory Service (NDS)</td>
</tr>
<tr>
<td>Domain Name System (DNS)</td>
</tr>
</tbody>
</table>
In order to solve these limitations and drawbacks, hence we propose the new naming system called Sharable Name System (SNS).

3. The New Naming System

3.1 System Environment

We consider the system environment based on most of organizations that have more than one office locations. For example, Prince of Songkla University (PSU) has 5 campuses located in different provinces that are HatYai, Pattani, Phuket, Surattani, and Trang. Figure 1 presents the structure of PSU and shows that each location consists of 3 divisions: Storehouse (S), Administration (A), and Human Resource (HR) divisions. Although these divisions have the same name, their locations are different. Each division of every location performs the same tasks. For example, the Administration division at HatYai has the same functions as Administration divisions of other 4 campuses. Using the same name for the same task helps everyone clearly understand and easily remembering, and also helps PSU maintains unity of its organizations. Therefore, the system environment that will be taken into the consideration of this research is an organization that consists of many branches with the same name and every single type of document uses the same document name as well. This situation is called “one name – many objects – one result” (ONMOOR).

3.2 Problem Statement

Solving the problem of “one name – many objects – one result” to serve the user needs is a challenging problem. Considering the situation of a person who need to contact a HR division of PSU or looking for a student information of PSU, the problem occurred is “which location (or document) is the right place (or document) for that person?”. Using a computer searching system like google.com, the user will obtain more than one answers at a search result, and the user may not be satisfied according to time consuming in searching for the right one.

Because of many naming systems are hierarchical structure and each node must be named differently. Therefore all existing naming systems cannot solve the problem stated above.

3.3 The Structure of SNS

The global name space is a hierarchical structure with a single root at the top. Each name is a path in the global name space. The objects are the global names that point to information about individual objects. A hierarchical structure of the SNS employs the concept of sets. Figure 2 illustrates the logic organization of the SNS in the sense of a global name able to be shared. A single root at the top in Figure 2 is named as “PSU” and other nodes in the tree are global names. A global name may logically identify not only a single node, but nodes in the tree. Each global name may point to information about individual object or a set of information about individual object. This means that the SNS maps each global name to a set of objects. For example, in Figure 2, the new structure provides a sharable name such a store.psu from the left branch would map storehouse at HatYai (obj1) and store.psu at the right branch maps storehouse at Pattani (obj2). However, if we call store.psu, the result provides a set of objects: [obj1, obj2].

3.4 The SNS Message Protocol

TCP/IP protocol suite contains standard application protocols for services. However, the existing standard protocols do not serve a need for sharing names. Therefore, we have designed a new protocol called SNS message protocol. The SNS queries and responses are often contained within TCP packets. Communication inside the SNS protocol is carried in a single format called an SNS message. All messages – both queries and responses – are divided into 3 sections: Header, Question, and Answer section as shown in Figure 3:

The header section contains fields specified whether a message is a query or response, and also specified which of the remaining sections are presented. The ID field is an 8-bit identifier. The fg is a 1-bit field that specifies whether this message is a query (0) or a response (1). The Opcode is a 3-bit field that specifies a kind of query in this message. The RCODE is a 4-bit field with the return code. The values have the following interpretation:

0 No error condition
1 Format error – name server cannot interpret the query.
2 Server failure – name server cannot process the query.
3 Not implemented – name server does not support the requested query type.
4-15 Reserved for future use.

The next two 16-bit fields specify the number of questions and the number of answers respectively. For a query, the number of question is normally 1 and the other field is 0. For a reply, the number of answer is at least 1.

The format of the question section is a query name. The query name is a global name being look up. It is a sequence of one or more labels. The query name field (QName) is stored in Unicode. Each Unicode character is encoded in UTF-8 (UCS Transformation Format). Each label is a length and a number of byte per character followed by the number of characters. The name terminated with a byte of 0. The following examples of global names show how they are stored.

A global name “store.psu” is stored as follow:

```
5.1 st o r e 3.1 p s u 0
```

The other global name is stored as follow:

```
4.3 ฮูทน 3.3 กอม 0
```

The last example shows a Thai global name which uses non-ASCII characters.

The global name that is able to stored in Unicode expands to encompass various languages not only English. Therefore, the limitations of scalability as we mentioned previously has been solved.

The answer section may contain one of more records called the SNS records. The number of records is specified in the corresponding field in the header section. Each SNS record contains name, time-to-live, SNS data length and SNSDATA field. Figure 4 describes format of an SNS record.

The name field is a global name which SNSDATA corresponds. The name field has a same format as described in query name field.

The time-to-live field (TTL) is the 16-bit field that specifies the time interval (in seconds) that the SNS record may be cached before it should be discarded. The SNS data length specifies the amount of SNS data.

The SNSDATA is a variable length string that describes the global name varies according to a set of objects including the two important attributes: location and created Date.

```
Name
Time-to-live
SNS data length
SNSDATA
```

**Figure 4 Format of SNS record.**

### 3.5 Query and Response example

The following query and response illustrates name server behavior. When a client queries a global name “store.psu” to a name server, the question section is stored store.psu. The query message format may look like:

<table>
<thead>
<tr>
<th>Header</th>
<th>Opcode = query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question</td>
<td>QName = store.psu</td>
</tr>
<tr>
<td>Answer</td>
<td>&lt;empty&gt;</td>
</tr>
</tbody>
</table>

The response may be:

<table>
<thead>
<tr>
<th>Header</th>
<th>QR = Response, Opcode = query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question</td>
<td>QName = store.psu</td>
</tr>
<tr>
<td>Answer</td>
<td>store.psu 86400 17 obj1 HatYai 13012001 store.psu 86400 18 obj2 Pattani 05032002</td>
</tr>
</tbody>
</table>

### 3.6 Active Components
SNS is a client/server model. The active components are name servers and resolvers. Resolvers are the clients that access name servers. Normally, a resolver is a part of an application. A resolver queries a name server to map names to objects. The concepts of sets and trees are provided for name to object translation. Each tree contains nodes which resolve to a set of objects or may refer to other servers. A path through the tree terminates at a node. The set of objects contained that in the node is returned as the result of the path. The querying type should be one of the following types: full path such as a global name; store.psu or a combination of global names; store.psu:warehouse.hatyai.psu, and some parts of the path such as store.psu:loc@hatyai. The path store.psu: warehouse.hatyai.psu is a set of paths.

Before we describe the mechanism of resolution process, let us take a closer look at the global name properties.

3.7 Global Name Properties

We use names to indicate the objects, and may refer as documented files, images, addresses, etc. Each name has its properties and certainly has an identity. Our work proposes a sharable name for objects, so that identifying the right object is very important issue. Therefore, we define a global class (Gclass) to store the information of each global object. Additionally, user class (Uclass) is defined to store some information such as location. We also show a relationship between the global class and the user class in Figure 5.

Figure 5 Class diagram of User and Global and their relationship.

Uclass stores one of the important attributes called Ulocality which is a value of client location. Gclass stores Gname, Gobj, Glocality, GcDate, and Gtype. Gname is a global name. Gobj is an object that related to Gname. Glocality is a value of a global object location. Every time a global name is queried, Gcount is increased by 1. Gcount will be set to 0 at the first time Gname is created or when time-to-live (TTL) is 0 or 1. Gtype attribute specifies whether a global name is a leaf node (Lf) or a node (Nd) in the global name space. Normally, a Ulocality attribute in Uclass comes from a location of a local name server that a client queries a global name. Each name server will store database of Gclass. Every time a new object is created, its information will be stored after registration process.

The example below refer to Figure 2.

Example : a local name server is psu and stores Gclass.

Suppose that Gclass contains six attributes: <Gname, Gobj, Glocality, createDate, Gcount, Gtype>. The following table shows the data stored in Gclass

<table>
<thead>
<tr>
<th>Gname</th>
<th>Gobj</th>
<th>Glocality</th>
<th>GcrDate</th>
<th>Gcount</th>
<th>Gtype</th>
</tr>
</thead>
<tbody>
<tr>
<td>store.psu</td>
<td>obj1</td>
<td>HatYai</td>
<td>13-01-2001</td>
<td>5</td>
<td>Lf</td>
</tr>
<tr>
<td>Store.psu</td>
<td>obj2</td>
<td>Pattani</td>
<td>05-03-2002</td>
<td>2</td>
<td>Lf</td>
</tr>
<tr>
<td>hatyai.psu</td>
<td>obj3</td>
<td>HatYai</td>
<td>12-11-1999</td>
<td>8</td>
<td>Nd</td>
</tr>
</tbody>
</table>

In addition, Uclass contains two attributes: <Uobj, Ulocality> and the example is shown below:

<table>
<thead>
<tr>
<th>Uobj</th>
<th>Ulocality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj1</td>
<td>HatYai</td>
</tr>
</tbody>
</table>

3.8 Resolution Process

To find the exact object, a resolution process has been constructed. When a client queries a name (q1, q2, ..., qn) which may be one of the previously querying types, at a local name server, and the local name server does not know the answer, a process to decompose the query into subqueries (q1, q2, ..., qn) is invoked. Other name servers are queried to determine a mapping from each subquery name to a set of objects. When each subquery returns, a process to intersect all subqueries is invoked and gives the answer (ANS) either a unique object or a set of objects. If ANS is a unique object, a local name server returns ANS to a client. If ANS is a set of objects, a local name server invokes a filtering process to guarantee a client to receive the right unique object based on a location or a latest created date.

The algorithm of FilteringProcess describes below:

1. Use Quicksort algorithm to sort ANS (Obj1, Obj2, ..., Objk) based on counting factors: location and created date, respectively.
2. A local name server takes a client location comparing to the location of each object in sorted ANS.
3. If one object (Objj) in sorted ANS has a same location as a client then
   Return Objj to a client
 else if there are a set of objects (Obj1, Obj2, ..., Objk) or no object in sorted ANS has a same location as a client then
   For each object in sorted ANS
   Find objj with a latest created date
   Return objj to a client.

This algorithm requires the expected time in O(n + n log n). So that, the complexity is O(n log n). The reason for choosing the latest created date object is based on the human nature that loves to get the new and up-to-date
object. However, if the answer after filtering does not satisfy the client, the user can ask for the whole answer list on a basis of hit. In this case, our hit comes from the Gcount attribute in Gclass. The more value in Gcount is, the more that object is queried.

Using the example of Gclass and Uclass described in section 3.7, the above algorithm gains the following results:

**Example**: Suppose a client at psu queries store.psu to its local name server “psu”, the ANS returns {obj1, obj2}. Therefore, the FilteringProcess is invoked and the final result is obj2 because no object in ANS has the same location as the client and the algorithm chooses the latest created date of the object in ANS and returns only one object back to the client.

4. Discussion

The SNS is based on TCP/IP protocol and provides facility for mapping one name to many objects. This feature fulfills a need of many organizations with many branches to maintain the unity of its organization and ease anyone to remember one name regardless of the objects are. As a result, we expect that the SNS can be implied to use not only in any organizations, but also in Internet such a referring to any web pages that may use the same name but different locations and FTP applications.

The structure of SNS employs both sets and trees. The property of tree helps the organization remains scalability. This means the increasing of the number of objects remains effective. Furthermore, the property of set provides one name can be referred to many objects. The filtering mechanism guarantees users to obtain the right objects. The time complexity of the filtering algorithm is O(n log n). The SNS message protocol that uses Unicode characters is designed to serve a need of naming objects both ASCII characters and non-ASCII characters. This property expands to encompass various languages. The SNS queries and responses are contained within TCP packets to guarantee that the data delivered between connections does not be lost, especially the SNS responses data.

5. Conclusion

The current name services are implemented in hierarchical structure. This structure contains a simple parent-child relationship which provides a unique name. Names at the leaf nodes of the tree represent individual objects. Thus, it limits names to be shared. This paper proposes SNS architecture to facilitate one name mapping many objects and obtains one result.

The new structure, on the other hand, employs both the concepts of sets and trees. So, names at the leaf nodes may not represent individual objects but a set of objects. The structure satisfies that one name can be shared. However, an exact object returns to a client is considered. A resolution process including a filtering process based on user location or created date and hits, has been proposed to find a required object.

We also consider to intensiveness of the resolution process which will increases traffic. So, caches maintained in the name servers will be used to reduce the traffic. Finally, using the proposed SNS message protocol, all limitations and drawback that mentioned in section 3 has been completely solved.

References