A Syntax of Universal NIC Names (UNIC)

Young S. Han*

목 차

1. Introduction

The elements underlying web services have never been fixed and have beenunder constant changes. The naming service that helps users access pages turned out to be not an exceptional part of such a violent evolutionary system. Naming services such as URL and domain names have critical deficiencies in so far as to cover the volume and complexity of today's web services.

The idea presented in this paper is a formal description over the universal naming service that is already under service in Korea. The objective of the new naming service led by KT (Korea Telecommunication Corporation) is to overcome a couple of barriers in dealing with online objects. The first one is to provide enough name space for all the objects on line and the second is to liberate the functional scope of names returning the regulatory rights to local name service owners.

The service name "UNIC(Universal Nickname)"is an embodiment of universal name service proposed by the author of this article. In the UNIC service led by KT, anyone can run one's own name service with a unique TLD (Top Level Domain) licensed by KT. The owner of a name service determines how the names are structured and what functions the names will be associated with. The UNIC office intends not to carry heavy regulations to be observed by the local authorities, and provides minimal specifications that cover the syntax of UNIC expressions and the protocols for the communication among local name servers.

The UNIC service is expected to make it much more efficient for companies to deliver their product information to the customers and to communicate with end users. The name scheme focuses on web pages instead of web sites, objects instead of categories, and contents instead of contents providers. The spirit of UNIC service is to give life to objects so that they may be distributed and communicated directly and independently of the object container to which they belong. The EPC (Electronic Product Code) stored in RFID and IPv6 for object level online address are footed on the common ground. In this regard, UNIC is the highest level of naming system for EPC, IPv6, and ENUM.

The UNIC could turn out to be a new system of web addressing by taking sufficiently different policies than the current domain system for the issues such as the authority of the naming services, the subjects of naming, and the flexibility of service extensions.

* 수원대학교 정보미디어학과 교수

In UNIC, the owner of a TLD can run a naming system with name syntax in his/her design and little restrictions and can add new functions to the names as long as the syntactic and semantic modifications may not interfere with other TLD's and the UNIC community as a whole. This makes UNIC depart from ordinary DNS architecture in which local DNS owners have few stuffs they can do.

The UNIC service may be available to the users through the supports from ISP's or the installation of browser plug in. Currently the service is now under service by the major ISP's in Korea (KT, Hanaro, and DACOM).

2. Motivations

UNIC may well be regarded as the descendant of URN (Universal Resource Naming) that was intended to overcome the volatility of URL's. URN and URL differ in that the former is a naming scheme and the latter is an addressing scheme. The problem of volatility with URL occurs because of the lower credibility of addresses that can change often. The names, on the other hand, for particular resources are less likely to change over the time. The access tags in names will last longer. UNIC also shares the similar service architecture with that of URN, but departs from URN by taking more aggressiveadditions such as followings:

1. to allow unlimited character sets
2. to allow non standard functions to the names
3. to maintain the Top Level Domain (TLD) idea of conventional domains

Such differentiation brings out new horizon of applications in contents distribution and ubiquitous computations.

2.1 unlimited space of TLD's

The UNIC (Universal NIC) is a naming scheme for objects and content units, for the UNIC's are attached to the objects on line, or on mobile. Offline objects are also named so long as they can be recognized and identified on line.

A name in UNIC may look like as follows.

(1) youngwoo.usw

In the example 1, "youngwoo" is a name for the subject belonging to the category "usw," which in this case is a university name. "usw"corresponds to a TLD in light of URL domains. In UNIC, a TLD is owned, operated, and regulated by local server, "usw," in the example. The idea that anyone can own a naming service with a TLD named afterhis/her choice was first proposed as early as July, 1999 by the author. Prototype service was introducedin Korea at the March of year 2000 by How&Why Company.

The expressive power of names in UNIC are rich enough to cover literally all the names in the world of objects on and off line and objects can be persons, web pages, multimedia contents, devices, articles, and books.

(2) product-name.company-name

A class of products or each product item can be given a name after the company name or the brand name of the item. The name can be used to advertisethe item and promote online and mobile purchases. When the name corresponds to serial number such as barcode or EPC (Electronic Product Code), the name is a verification of products and a gateway to A/S exclusively for the item.

(3) the syntax of UNIC.paper

The title of a paper can be an access identification for direct references. One need not worry about

recovering a paper on line with the title of the paper known.

(4) refrigerator.ywlee

Even more specific application can be a case in which the UNIC name is used to designate a particular device or machine for control. The above name when issued with control arguments can make the refrigerator owned by ywlee change its status. Alternatively the name may be linked to a control page for the operation of the machine.

As is shown in the above, the names in UNIC are far much more spacious than conventional domains canoffer, for TLD domain is made to serve as identifiers just like 2nd level domains in URL domains.

2.3 Functional Flexibility

In additionto the extension of naming space, UNIC allows the authority of a naming system to incorporate arbitrary functions into names. In other words, local semantics can be implemented so long as they donot cause problems with known functions.

In fact URL's are associated with web pages or web functions, and the semantics of domain names vary according to the protocols in which the names are used. Those protocols such as ftp and telnet are standardized and are not in the scope of the naming authorities.

URL's are powerful, but hard to remember and communicate through human memories. Domain names are easier to remember, but rigid in semantics. UNIC's approach is to have the names mean anything the service providers or the name owners want.

As an example of proprietary functions, one may encode functions into the names.

(5) check:youngkim05.usw

The above string may mean that the user wants to check the enrollment status of student "youngkim05" in the suwon university.

As another example, names can carry arguments to execute proprietary functions.

(6) youngkim05.usw\vote 10

where '\' denotes separation between UNIC and its functional arguments.

One can imagine that a UNIC user makes a popular vote for candidate youngkim05. This may sound a bit extreme. Declarative approach to web service like the one shown in the above may be advantageous in some areas. Clear communication is one benefit over the URL or web page based services, for one must have lengthyURL for a service and be patient to go through readings and multiple clicks to complete a service of interest.

As one might insist, the biggest shortcoming of the proprietary functions is that one must remember the syntax and semantics of each UNIC service. There are few people who would remember URL's and there will be no need of remembering the numerous details of UNIC functions for many services will be available to them. UNIC functions may be available to the members of closed communities and the members are likely to memorize the function usages or be guided by manuals and handouts. It is for sure simpler to communicate the UNIC functions with others than to deliver web pages that can only be understood procedurally.

There are fourstandard schemes in UNIC as of this day. Local service operators can implement proprietary schemes of their own by defining further structures in name strings or by using arguments.

scheme 1 : resource access

The UNIC itself represents a resource stored in a particular server, thus without further arguments the name will guide the user to the resource.

[UNICexpression]

The UNICexpression is prescribed in the later part of the paper.

scheme 2 : message delivery to the resource unit

The idea is that every object can be a message recipientand it is not just a subject to be accessed but also an entity to communicate with. In reality it is assumed there are corresponding persons behind every object to listen to the messages arrived at the object.In another assumption, machines may receive messages and act accordingly.

The syntax of the scheme is as follows.

[UNICexpression/[blank] message content]

One may enter the name of message recipient in UNIC and "?" followed by message string onto the address bar of explorer browser, expecting the message will be delivered to the mobile phone or email box of the subject uniquely identified by the name.

UNIC is convergent and universal for one more reason from the message scheme. The idea behind the message scheme is identification system can be unified and thus can be multi modal being multifunctional in any interfaces.

scheme 3 : subordinate resource access

A UNIC name may represent a set of resources, and each of them can be specified after UNIC name as follows.

[UNICexpression/resourcename]

This scheme is anything but new as the semantics is common in URL expressions.

scheme 4 : proprietary functions

The authority of naming and resolver management can implement non standard functions requiring arguments. The delimiter in this case is "\" followed by functional specifications.

[UNICexpression\functionname: arguments]

The syntax after delimiter "\" is arbitrary and is up to the authority.

3. Syntactic Description of UNIC

The surface form of UNIC names take after that of URL domains, for '.' is used as a delimiter between domains and the right most domains are the most significant ones (TLD equivalents). Due to the similarity, TLD's may cause confusion, and the TLD's by ICANN (Internet Corporation for Assigned Names and Numbers) will not be reassigned by theUNIC communities. The primary scope of UNIC is on the space of objects not web sites, thus it is possible to view UNIC and ICANN complement each other.

The world of objects and contents requires more expressive power for their name tags, thus the left hand side of UNIC need to be allowed to use symbols as well as characters and numbers.

UNIC names containing symbols are useful as follows.

(7) peter&michael.bud

(8) $100sale.sears

(9) E=MC^2.physics

The liberty of composing UNIC names is not something completely unlimited, however. Some UNIC standard services prohibit the use of particular symbols or symbol sequences. The term UNIC representsa name expression. Applied expressions or protocol expressions includes UNIC along with appropriate arguments. The UNIC service expression consists of two components : UNIC name and UNIC argument(s). The name part is separated by the rightmost dot (.) and left side of the dot is for naming objects and the right side is area for TLD's.

<UNIC Service> := <object-name>
.<TLD><del><arguments>

As is shown in the above example, a full expression resembles a command function in which [object-name] corresponds to a function name, and this illuminates the uniqueness of UNIC service concept.

```
<UNICSVC>       ::= <UNIC> <del> <ARG>

<UNIC>          ::= <leftunic> "." <namespace>
<ARG>           ::=  [1,1024 <let-num-sym3>] -
symbolseq
<leftunic>      ::= <leftunic> "." <namespace> |
    <let-num> [ 1, 127 <let-num-sym> ]
<namespace>     ::= <let-num> [ 1, 63 <let-num-sym2> ]
<let-num>       ::= <num> | <kalpha> | <ealpha>
<let-num-sym>   ::=  <num> | <kalpha> | <ealpha> |
                     <sym>
<let-num-sym2>  ::=  <num> | <kalpha> | <ealpha> |
                     <sym2>
<let-numsym3>   ::=  <num> | <kalpha> | <ealpha> |
                     <sym> | <sym2>
<num>           ::= "0" | "1" | ... | "9"
<ealpha>        ::= "A" | "a" | "B" | "b" | ... | "Z" | "z"
<kalpha>        ::= "ㄱ" | "ㄴ" | ... | "ㅎ" | "가" |
                    "각" | ... | "힝"
<sym>           ::= "(" | ")" | "+" | "," | "-" | "#" |
                    "^" | "\" |"~" |"`" | "" " | "&" |
                    "=" | ";" | "$" | "[" | "]" | "{" |
                    "}" | "<" | ">" | "?" | "/" | "|" |
                    "_" | "!" | "*" | "a"
<sym2>          ::= "-" | "=" | "+" | "_" | "(" | ")" |
                    "[" | "]" | "$" | "*" | "&" | "^"
<del>           ::=  " "  |  "/" |  "\"
<symbolseq>     ::= [1, 127 <hyphen> ">" ] |  [1,
                    127 <arrow> ">" ]
<hyphen>        ::=  <hyphen> "-" | "-"
<arrow>         :: =  <arrow> ">" | ">"
```

As to the symbol selection, the limitations by the existing services were acknowledged. For instance, the symbol set in <leftunic> includes almost all the symbols except those that are central in other applications. They include ":", "%", "@", and "'". Dots (".") are also a primary delimiter in UNIC just as in URL's. Blanks are allowed in <leftunic> which corresponds to the name of object. Once first dot is found, <namespace> is expected. There can be more than one namespaces in which case except the rightmost one the authority of naming and management of namespaces belongs to local service owners.

It, however, must be noted that the liberality of symbols in UNIC is reduced to the bounds set by the web browsers. No symbols except '-' would be allowed in the name parts in the existing browsers. Consequently the working version of UNIC implemented in Korea does not allow the symbols as in the above.

The message syntax is worth mentioning. The following are valid messaging expressions.

(10) peterlee.lawyer/ hello mr. lee, give me a call

In this message expression, the recipient is "peterlee.laywer"who is presumed to be a lawyer. The message sender is not explicitly scribed. It is up to the capability of resolution server for ".lawyer"how the sender is going to be recognized. The resolution server may request the sender for identifying himself/herself.

The message receiver can access the message through any medium the receiver wishes. For instance, it can be SMS, email, or messenger message.

5. Implementation Issues

There are many topics in realizing UNIC services. As URN (Universal Resource Name) has been studied over the years under the topics of functional requirements (Sollins et al, 1994), syntax (Moats, 1997), and resolutions (Sollins, 1996, 1998), UNIC may need comparable volume of discussions and efforts.

As far as the spirit of URN is concerned, UNIC is a

unique composite of URN and HFN (Human Friendly Name). URN is meant for system level applications and thus for human clients extra mechanism to translate natural language expressions into URN expressions is needed (Sollins, 1998). UNIC does not presume intermediary level such URN's and relate HFN level with resource addresses directly.

In the following issues are briefly mentioned with respect to those of URN.

Design Principles

In laying out UNIC, functional requirements are mostly congruent with those of URN (Sollins, 1994).

The scope and uniqueness of UNIC must be globally observed with consistency.

The resource names may well be persistent as long as the lifetime of the resources. This property is recommended, but is not required.

The namingauthority of a particular namespace and the management of a resolving server can be delegated, and in fact the two authorities can be different where they can be also the same.

Resolution Service Architecture

Resolution service is central to the way UNIC is presented to the relevant users. The current resolution network is organized with one central server (Root Server) and many local resolvers. The resolvers are closely interconnected and support services of one another.

Also what arises in reality to implement an addressing service is how to upgrade the web browser or DNS servers. For commercial point of view, UNIC needs to be automatically supported by the web browsers without additional client program, and this suggests DNS services be modified. For practical reasons, client program is also prepared as all the DNS servers cannot be modified at once and there are many interfering programs.

6. Conclusion

One may conclude UNIC is a commercializedversion of URN with several advancements. The differential points are outline in the following.

- unlimited character sets

The character sets are not limited to support internationalization as is supported in domains (Hoffman, 2000)

- messaging with names(convergence and multi modality)

Known protocols associated with URN and URL are access related in nature. UNIC is added by a powerful concept of convergence of accessing and messaging through a name.

- simplified ownership of authorities

Naming authority and the management of resolving server ratherbe the same and they are solely responsible for the service design in their namespace. Very distinct services associated with existing web services are possible.

- uniform syntax with URL domains

The syntax of UNIC resembles that of URL domains to have UNIC services presented to the naïveusers as simply as possible without making them learn a new language.

UNIC helps shape new ways of information access and that will change the way web sites and pages are built. The users can now access a product page directly and let the page flow to others easily. Accessing and searching for resources are all done at the address bar and even messages can be sent to everything that will spawn a new breed of applications in e-commerce and community services.

Message architecture as well as resolution network, many design issues, and semantic description will be covered in later

References

[1] [Moats97] Moats. URN Syntax. RFC 2141. 1997.

[2] [Sollins94] Sollins. Functional Requirements for Uniform Resource Names, RFC 1737. 1994.

[3] [Sollins98] Sollins. Architectural Principles of Uniform Resource Name Resolution. RFC 2276. 1998.

[4] [Hoffman2000] Paul Hoffman.Preparation of Internationalized Host Names, draft-ietf-idnnameprep-00.txt. 2000.

◐ 저 자 소 개 ◑

한 영 석

1987년 오클라호마 주립대 전산학과 학사
1989년 오클라호마 주립대 전산학과 석사
1995년 한국과학기술원 전산학과 박사
1995년 연구개발정보센터 연구원
1996년 ~ 현재 수원대학교 정보미디어학과 교수
현재 수원대학교 정보미디어학과 교수 (1996~)