논리적 URL 사용을 위한 포워딩 서버의 설계 및 구현

양희재*

Design and Implementation of a Forwarding Server for Using the Logical URL

Heejae Yang*

요 약

웹이 인터넷 정보도구의 표준으로 자리잡게 됨에 따라 점점 더 많은 웹 사이트들이 만들어졌고, 동시에 더 많은 웹 주소 공간이 필요하게 되었다. 그러나 웹 주소를 나타내는 물리적 URL은 기억하기에 너무 길 뿐 아니라 웹 사이트 내용을 직관적으로 나타내기 어려운 점이 많았다. 본 논문에서는 물리적 URL 대신 논리적 URL을 사용하여 이런 어려움을 개선할 수 있도록 하는 URL 전달 서버의 구현에 대해 고찰하였다. URL 전달서버는 HTTP 재방향 트랜잭션을 이용하는 도메인 매퍼와 BIND에 기반한 네임서버로 구성되어지며, 본 논문에서는 이들 상호간의 작용 및 동작에 대해 상세히 연구하였다. 실험 결과 URL 전달에 따른 시간은 현재의 인터넷 트래픽 조건에 비해 무시할 만한 수준인 것으로 나타났다.

ABSTRACT

A lot many WWW sites are come into the world more and more as Web is considered as the unified Internet information tool. The location of each site or resource is usually specified by a physical URL, which is often too long to remember and tends to raise difficulty to show the aim of the site intuitively by seeing it. Since any person or organization can get his/her own domain name easily, it is more desirable to use a logical URL with the domain name which can be chosen more compact to remember and meaningful to represent the ultimate intention of the site.

This paper presents an implementation of a URL forwarding server which forwards a URL to another, so that a WWW site can use a logical URL instead of a physical one. The server consists of a domain mapper which uses the redirection transaction of the HTTP protocol, and a name server based on the BIND. The paper shows how the interaction between the domain mapper and the name server can make forwarding possible and describes its implementation in detail. Experimental results shows that the overhead incurred by URL forwarding is negligible compared to the typical delay of current Internet traffic condition.

키워드

DNS, Name Server, URL forwarding, WWW

Ⅰ. 서 론

인터넷 상의 모든 호스트 컴퓨터들은 자신만이 갖는 유일한 주소, 즉 IP 주소를 갖는다. 이 주소는 32비트의 크기를 갖는 이진수이지만, 사람들의 기억을 쉽게 하기 위해 8비트씩 나누고 각각을 사람이 친숙하게 느끼는 십진수로 바꾼 형태로 표기하고 있다. 예를 들어 11.22.33.44 등과 같은 주소가 있을 수 있다.

그러나 사람들은 IP 주소가 비록 십진수 값을 갖는다고 하더라도 여전히 그것을 기억하는 데 어려움

*경성대학교 컴퓨터공학과

접수일자 : 2003. 2. 26

을 느끼며, 또 숫자 자체로는 아무런 의미 부여도 할 수 없는 등의 이유로 인해 새로운 인터넷 주소 체계를 개발하였는데, 그것이 바로 도메인명 체계(Domain Name System)이다. 숫자로 구성되는 IP 주소와 달리 도메인명은 인터넷 호스트 주소를 더욱 의미있고 기억하기 쉽도록 기호로 나타낸 것이다[1][2][3].

기억상의 편리 외에도 도메인명은 인터넷 주소 공간을 크게 확대시킬 수 있다는 장점을 가진다. IP 주소는 32비트의 한정된 수로 구성되었기 때문에 주소 공간의 크기가 유한한데 비해, 도메인명은 임의의 문자 이름을 부여할 수 있으므로 주소 공간의 크기가 거의 무한대에 이른다.

하나의 IP 주소에 대해 복수개의 도메인명을 부여할 수 있다는 것은 도메인명 체계의 큰 장점이다. 도메인명은 궁극적으로 IP 주소로 매핑(mapping)되어지는데, 이런 매핑을 담당하는 것이 바로 네임서버(Name Server)이다. 이 매핑은 반드시 1:1일 필요는 없으며, 도메인명의 개수를 n이라 할 때 임의의 n에 대해 n:1의 매핑이 가능하다. 예를 들어 11.22.33.44라는 IP 주소를 갖는 도메인명은 www.company.com이 될 수도 있고, www.organization.org이 될 수도 있다. 현재와 같이 IP 주소가 고갈되어있는 상황에서 도메인명의 효과적인 사용은 인터넷의 활용도를 더욱 늘리는데 큰 역할을 할 수 있을 것이다.

웹의 폭발적 사용 증가에 따라 웹 사이트의 개수도 급격히 증가하고 있다. 웹 서버의 숫자가 비교적 제한적인데 비해 요구되는 웹 사이트의 수는 훨씬 많기 때문에 하나의 웹 서버에 여러개의 웹 싸이트를 구축하는 경우가 일반적이다. 예를 들어 http://www.company.com에는 그 기관의 대표적인 홈페이지를 두고, http://www.company.com/~kim에는 그 기관에 속해있는 kim 이라는 사용자의 개인 싸이트를 두는 등 하나의 웹 서버에 복수개의 웹 싸이트가 존재한다. 또한 kim이라는 사용자는 자바 관련 전문 싸이트를 역시 자신이 속한 기관의 호스트 컴퓨터에 둘 수 있다. 이 경우 그 싸이트의 주소는 http://www.company.com/~kim/java와 같이 될 것이다.

여기서 우리는 웹 주소를 나타내는 위치지정자, 즉 URL (Uniform Resource Locator)에 문제점이 있음을 발견할 수 있다. 웹 주소를 위한 URL의 일반적 구성은 앞의 예와 같이 http://host.domain/path와 같이 이루어진다[4]. 쉽게 기억될 수 있는 웹 싸이트 주소를 위해서는 URL 명이 간결하면서도 의미있는 이름을 가져야 한다는 것은 필수적 요건이 아닐 수 없다. 그러나 일반 사용자들이 이런 요건을 갖춘 URL 명을 가지기는 어렵다. 대개의 경우 도메인명은 자신이 속해있는 기관 또는 부서에 의해 미리 정해져 있거나, 또는 자신이 가입해 있는 ISP에 의해 결정되어진다. 앞의 예에서 kim이라는 사용자는 자신이 원하든 원하지 않든 자신의 개인 웹 싸이트 주소에 회사명이 들어갈 수 밖에 없게 된다. 즉 자신을 나타내는 고유한 주소라기보다는 특정회사에 소속된 개인이라는 의미가 강한 URL을 부득이 갖게 되는 것이다. http://www.company.com/~kim/java의 경우도 마찬가지이다. 회사나 자신을 나타내지 않고 오직 자바 관련 전문 싸이트로 이 웹 싸이트를 운영하고 싶지만 URL에 회사명과 사용자명이 모두 나타나므로 자바라는 주제만을 부각시키기가 어렵게 되는 것이다. 또한 이 URL은 그 길이가 너무 길기 때문에 외우기가 어렵다는 단점도 있다.

이런 문제점들은 모두 물리적 URL의 사용에 기인한 것이다. 원래 물리적 URL의 목적은 특정 자원이 놓여있는 인터넷 상의 실제 위치를 알려주는 역할을 하는 것이다. 그러나 대부분의 웹 이용자들에게는 실제 위치를 알려주는 물리적 URL 보다는 그 자원의 의미를 보다 잘 나타내는 논리적 URL이 더 중요하다. 앞서 말한 바와 같이 도메인명은 IP 주소와 달리 한 호스트에 대해 여러 개의 명칭이 부여될 수 있으므로 kim이라는 사용자의 URL에 반드시 www.company.com이라는 명칭을 포함시킬 필요가 없다. 즉 kim은 자신의 URL을 http://www.company.com/~kim으로 할 것이 아니라 원할 경우 http://www.kim.com으로 할 수도 있으며(이 경우 더 이상 특정회사에 속한 개인이라는 의미를 나타내지 않게 된다), http://kim.company.com으로도 할 수 있다는 것이다. 두 번째 예의 자바 관련 전문 싸이트의 URL도 http://www.company.com/~kim/java라고 할 것이 아니라 http://www.java.org와 같이 할 수 있다. 앞의 경우는 물리적 URL, 뒤의 경우는 논리적 URL이 된다. 논리적 URL에 포함되어있는 도메인명을 가상 도메인명(virtual domain name)이라고 부른다.

논리적 URL은 궁극적으로 물리적 URL 위치로

전달되어지며, 이것을 URL 전달(URL forwarding)이라고 한다. 본 논문에서 설명하는 URL 전달의 핵심은 HTTP 규격에 정의되어 있는 방향재설정(redirection) 기능과, URL 에 포함되어있는 도메인명을 IP 주소로 변환하는 네임서버의 활용이다[3][5][6]. 여기서는 HTTP 방향재설정을 위한 웹 서버와 네임서버의 상호작용을 통해 URL 전달이 이루어지는 과정에 대해 고찰해 보고, 효율적 URL 전달을 위한 URL 전달 서버의 구현에 대해 알아보고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 URL 전달을 위한 이론적 배경에 대해 알아보고, 3장에서는 HTTP 규격에서 정한 방향재전송 기능, 네임서버의 환경설정, 이메일 전달 등에 대해 설명한다. 4장에서는 이런 배경을 바탕으로 실제 구현에 대해 설명하며, 5장에서 관련연구에 대해 기술하였다. 6장은 결론 및 향후 연구방향에 대해 정리하였다.

II. URL 전달

2.1 웹 서버와 클라이언트의 상호작용

웹은 원격지 서버에 놓여있는 하이퍼텍스트 문서, 영상, 소리 등의 자료를 효과적으로 접근할 수 있도록 개발된 인터넷 정보도구로서, 웹 브라우저(즉 클라이언트) 프로그램을 사용하여 서버의 정보를 이용하는 전형적인 클라이언트/서버 환경의 시스템이다.

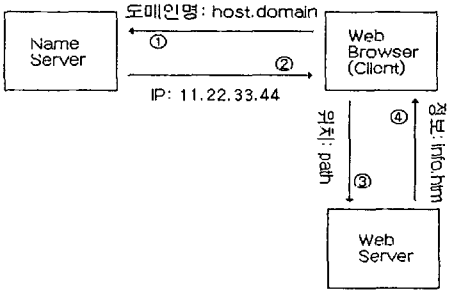

그림 1. 웹 서버 접근 과정
Fig. 1 Procedure of accessing a Web server

웹 서버와 웹 클라이언트 사이의 정보전달은 HTTP 전송규약에 따른다. 웹 서버 상에 놓인 정보들은 http://host.domain/path의 형식을 갖는 URL로 나타내어지며, 클라이언트는 도메인명으로 표시되어 있는 웹 서버의 실제 IP 주소를 얻기 위해 네임서버에 먼저 접근하고 이후 그 IP 주소를 갖는 웹 서버에 접근한다. 그림 1은 웹 클라이언트가 웹 서버에 놓인 정보를 얻는 일련의 과정을 보여준다.

2.2 URL 전달

URL 전달은 http://host.domain/path 로 표기되는 물리적 URL 대신 http://virtualhost.virtualdomain/virtualpath로 표기되는 논리적 URL을 사용하여 동일 정보에 접근할 수 있게 하는 것이다.

여기서 우리는 URL 전달을 위해서 두가지의 매핑이 이루어져야 함을 알 수 있다. 즉 가상 도메인명이 실제 도메인명으로 매핑되어야 하며(virtualhost.virtualdomain → host.domain), 가상 경로명이 실제 경로명으로 매핑되어야 한다(virtualpath → path).

(1) 가상 도메인명의 매핑

먼저 가상 도메인명의 매핑에 대해 알아보자. 모든 도메인명은 궁극적으로 IP 주소로 변환되어져야 한다. 즉 웹 브라우저가 지정한 URL에 포함된 도메인명은 그림 1에서 보인 바와 같이 네임서버에 의해 IP 주소로 변환되며, 웹 브라우저는 해당 IP 주소를 갖는 웹 서버에 접근하게 되는 것이다. 이것은 실제 도메인명이나 가상 도메인명이나 구분없이 동일하게 적용된다.

그러나 가상 도메인명에 해당되는 IP 주소를 얻는 방법은 1) 네임서버로부터 직접 가상 도메인명에 해당되는 IP 값을 얻는 방법과, 2) 가상 도메인명을 실제 도메인명으로 변경하고, 그후 다시 실제 도메인명에 해당되는 IP 값을 네임서버로부터 얻는 등의 두가지 방법이 있을 수 있다(그림 2).

먼저 첫번째 방법에 대해 알아보자. 그림 2(a)에서 보인 것과 같이 지정된 URL 내의 가상 도메인명 vhost.vdomain의 IP 주소를 얻기 위해 브라우저는 먼저 네임서버에게 접근한다(①). 네임서버에게 vhost.vdomain이라는 도메인명을 넘겨주면 네임서버는 자신의 내부 테이블을 검색하게 되는데, 만일 이 테이블 내에 host.domain이라는 실제 도메인명의 IP 주소와 vhost.vdomain이라는 가상 도메인명의 IP 주

소를 같은 값으로 넣어 주었다면 네임서버는 그것의 IP 주소인 11.22.33.44를 브라우저에게 넘겨준다. 브라우저는 이 IP 주소를 사용하여 실제 웹 서버에 접근하게 된다.

다음으로 두번째 방법인 그림 2(b)에 대해서 알아보자. 여기서는 네임서버 외에 도메인 매퍼(Domain Mapper)라는 서버를 두어 이것으로 하여금 가상 도메인명을 실제 도메인명으로 변경하게 하는 역할을 하게 한다. 앞에서와 마찬가지로 지정된 URL 내의 가상 도메인명의 IP 주소를 얻기 위해 브라우저는 일반적인 경우와 같이 먼저 네임서버에게 접근한다(①). DNS가 vhost.vdomain의 IP 주소를 99.88.77.66이라고 알려주면(②) 브라우저는 이 IP 주소를 갖는 가상의 웹 서버 컴퓨터에 접근하여 HTTP 규격에 따라 각종 요청정보(request information)를 보내게 되는데, 이 요청정보 내용 중에는 HTTP 규격에 따라 vhost.vdomain라는 이름이 포함되어 있다(③). 요청정보를 받아들인 웹 서버 컴퓨터는 역시 HTTP 규격에 따라 응답정보(reply information)를 보내게 되는데, 우리는 이 응답정보 중의 하나가 가상 도메인명 vhost.vdomain의 실제 도메인명인 host.domain이 되도록 한다(④). 즉 이 웹 서버 컴퓨터가 도메인 매퍼(domain mapper)의 역할을 하게 하는 것이다.

도메인 매퍼로부터 실제 도메인명 host.domain을 얻게 된 브라우저는 다시 이것의 IP 주소를 얻기 위해 DNS에게 한 번 더 접근하게 되고(⑤), DNS로부터 IP 주소(예를 들어) 11.22.33.44를 얻은 후(⑥), 최종적으로 이 IP 주소를 갖는 실제 웹 서버에 접근하게 되는 것이다.

이 두 번째 방법은 따라서 실제 웹 서버에 접근하기 위해 IP 주소를 찾는데만 ①번부터 ⑥번까지 모두 6단계를 거쳐야 된다. 가상 도메인명을 사용하지 않는 일반적 경우(그림 1) 불과 2단계로 IP 주소를 얻을 수 있는 것에 비해 매우 큰 추가부담을 필요로 함을 알 수 있다. 별도의 도메인 매핑을 위한 웹 서버를 마련해야 하며, 네임서버에 대해서도 두 번의 접근을 필요로 한다. 반면 첫 번째 방법은 네임서버에 단지 host.domain 외에 vhost.vdomain 항목을 추가하여 그들의 IP 주소를 동일하게 두는 것에 지나지 않으므로 IP 주소를 얻는데 따른 부담은 가상 도메인명을 사용하지 않는 경우와 마찬가지로 2단계에 불과하다.

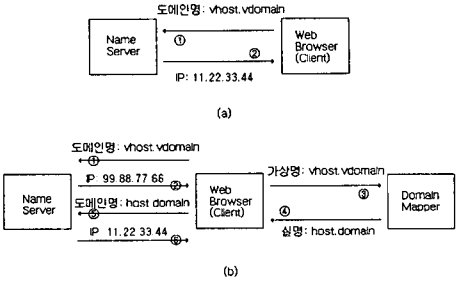

그림 2. 가상 도메인명의 사용
Fig. 2 Use of a virtual domain name

(2) 가상 경로명의 매핑

논리적 URL http://virtualhost.virtualdomain/virtualpath의 내용 중 virtualpath 부분이 가상 경로명에 해당되며, 이것은 실제 URL http://host.domain/path의 내용 중 path 부분에 해당되는 실제 경로명으로 매핑되어야 한다. 서론의 예와 같이 http://www.kim.com(실제로는 http://www.kim.com/index.html)이 http://www.company.com/~kim(실제로는 http://www.company.com/~kim/index.html)으로 전달되기 위해서는 www.kim.com의 IP 주소가 www.company.com의 IP 주소와 같아야 하며, /index.html이 /~kim/index.html와 같아야 한다.

가상 도메인명의 매핑과는 달리 경로명의 매핑에는 네임서버의 개입이 없다. 따라서 virtualpath로부터 path로의 매핑은 네임서버 외의 다른 곳에서 이루어져야 하는데, 그 후보로는 실제 웹 서버 컴퓨터 또는 그림 2(b)에서 제안된 도메인 매퍼가 있다.

먼저 실제 웹 서버 컴퓨터를 사용하는 방법에 대해 알아보자. 그림 2(a)와 같은 구조의 경우 virtualpath로부터 path 로의 매핑은 오직 host.domain에 해당되는 웹 서버에서 이루어질 수 밖에 없는데, 이같은 매핑을 하기 위해서는 해당 웹 서버의 환경설정이 미리부터 이 매핑을 포함하고 있어야만 한다. 대개의 경우 웹 서버의 환경설정은 해당기관의 시스템 관리자(즉 수퍼유저)만이 할 수 있으며, 이 시스템의 자유로운 사용에는 규제가 따르므로 일반

```
GET /vpath HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)
Host: vhost.vdomain
Connection: Keep-Alive
```

그림 3. 웹 브라우저가 보내는 요청정보
Fig. 3 Request from Web browser

개인의 경우 불편하기도 하며 또한 비현실적인 것이 되기 쉽다. 대부분의 회사 등 기관에서는 자사의 개인 사용자를 위해 웹 서버의 환경설정을 임의로 바꾸는 것을 허락하지 않기 때문이다.

도메인 매퍼를 사용하는 그림 2(b)의 경우는 훨씬 많은 유연성을 가질 수 있다. 앞에서 말한 바와 같이 ③번 과정에서 브라우저가 도메인 매퍼에게 HTTP 규격에 따른 요청정보를 보내는데, 이 요청정보에는 vhost.vdomain이라는 정보 외에도 가상경로명, 즉 vpath가 함께 포함되어 있다. 따라서 도메인 매퍼의 내부 매핑 테이블에 vpath를 path로 바꾸어주도록 하는 내용을 넣어두면 ④번 과정의 응답정보에 매핑된 vpath가 포함된 정보, 즉 vhost.vdomain/vpath가 브라우저에게 전달될 수 있다. 브라우저는 vhost.vdomain에 해당되는 IP 주소를 네임서버에게서 얻고, 다시 그 IP 주소를 갖는 웹 서버에 접근하게 되는데 이때 vpath 정보를 웹 서버에게 넘겨주므로 원하는 매핑이 이루어지게 되는 것이다. 이런 일련의 과정은 브라우저와 네임서버, 그리고 도메인 매퍼 사이에서 감추어져서 일어나므로 사용자들은 투명한 URL 전달 효과를 얻을 수 있으며, 실제 웹 서버에는 아무런 변경이 없으므로 시스템 관리자가 별도로 환경설정을 바꿀 필요도 없다.

이상의 이유로 우리는 URL 전달 방법으로 그림 2(b)와 같은 방식을 채택하였다. 이 방법은 별도의 도메인 매퍼를 두어야 하는 부담도 있으며, 네임서버를 두 번 접근하여야 되는 등의 이유로 속도도 늦어지지만 URL 전달을 위한 유연성이 훨씬 크다는 장점이 있다. 또한 4장에서 나타내었지만, 실제로 웹 페이지를 읽는데 소요되는 추가 시간은 불과 8msec 내외로 현재의 인터넷 전송속도를 감안할 때 큰 문제가 되지는 않았다.

III. URL 전달 서버의 설계

이 장에서 우리는 URL 전달을 위한 서버의 상세 설계에 대해 소개하고자 한다. 본 논문에서는 그림 2(b)와 같이 별도의 도메인 매퍼를 사용하는 방식을 채택하였다.

3.1 도메인 매퍼

그림 1에서 보인 바와 같이 웹 브라우저는 임의의 URL http://host.domain/path를 접근하기 위해 먼저 host.domain에 해당되는 IP 주소를 DNS로부터 얻고, 이 IP 주소에 해당되는 웹 서버에 연결을 시도한다. 본 논문에서 제안하는 URL 전달의 핵심은 이 웹 서버를 도메인 매퍼로 치환시키고, 이 도메인 매퍼는 HTTP 규격에 따른 재방향설정(Redirection) 응답 메시지를 브라우저에게 전달함으로서 브라우저가 물리적 URL을 찾아가게 한다는 것이다.

HTTP 규격에 의하면 웹 클라이언트와 웹 서버 사이의 작용은 요청 및 응답 (request & reply)에 의해 이루어진다. 웹 클라이언트, 즉 웹 브라우저가 논리적 URL http://vhost.vdomain/vpath에 해당되는 요청을 내면 웹 서버에게는 그림 3과 같은 요청정보가 보내지게 된다[5][6].

사용하는 브라우저의 종류 및 환경에 따라 약간의 차이는 있을 수 있다. 여기서 가장 중요한 부분은 요청정보의 제일 첫줄에 있는 GET 이하의 내용이다. GET은 HTTP에서 사용되는 여러 가지 메소드 중에서 가장 대표적인 것이며, 그 다음에 따라오는 /vpath에 해당되는 정보를 읽겠다는 뜻이다. 뒤의 HTTP/1.1은 사용하는 프로토콜이 HTTP 버전 1.1이라는 것을 의미한다.

동일하게 중요성을 갖는 요청정보 내용은 중간 부근의 Host: 필드이다. 이 필드에는 이 브라우저가 접속하고자 하는 서버 컴퓨터의 도메인명인 vhost.vdomain이 포함된다. 이 값은 IP 주소가 아니라 기호를 사용하는 도메인명임에 유의하라.

이러한 브라우저의 요청정보에 대해 웹 서버는 그림 4와 유사한 응답정보를 넘겨준다. 이와 같은

응답정보는 웹 서버의 디스크 내에 브라우저가 요청한 /vpath라는 자원이 존재하는 경우에 보내어지는 것이다. 제일 첫째줄의 HTTP/1.1은 사용하는 HTTP 버전번호를 나타내며, 그 뒤 따라오는 200 은 응답코드로서 정상적인 트랜잭션이 이루어졌음을 의미한다. 그 뒤의 OK는 디버깅 목적을 위해 사람들이 참조할 수 있도록 덧붙인 것이다.

HTTP 규격에 따르면 응답코드의 종류는 정상적인 트랜잭션을 알리는 200 외에도 여러 가지가 있다[5][6]. 응답코드는 세 개의 자리수를 갖는 십진수로 이루어지는데, 이 값의 첫 자리가 2로 시작되면 정상, 4 또는 5로 시작되면 트랜잭션이 실패한 것을 의미한다. 응답코드가 3으로 시작되는 것은 브라우저가 요청한 자원이 현재의 URL로 나타낸 위치에 있지 않고 다른 곳으로 임시 또는 항구적으로 옮겨졌음을 의미하는 재전송(redirection) 트랜잭션을 말한다. 본 논문에서 제시하는 URL 전달 서버는 바로 이 재전송 트랜잭션을 이용한 것이다.

재전송 트랜잭션이 일어나는 경우 응답정보는 다음과 같은 형태를 가진다(그림 5).

따라서 new_URL에 해당되는 내용을 실제 자원이 존재하는 위치, 즉 http://host.domain/path로 두면 브라우저는 이 응답정보를 받고 다시 host.domain 에 해당되는 웹 서버에게 /path에 해당되는 자원을 요청하게 되는 것이다. 즉 vhost.vdomain에 해당되는 웹 서버로 하여금 브라우저가 원래 요청한 자원이 다른 웹 서버의 다른 경로, 즉 host.domain/path에 있다는 응답을 주게되면 자동 재전송이 일어나게 되는 것이며, 이 경우 vhost.vdomain에 해당되는 웹 서버는 더 이상 웹 서버가 아니라 도메인 매퍼, 즉 재전송 서버의 역할을 하게 된다.

도메인 매퍼의 입력으로 주어지는 정보, 즉 브라우저가 보내 온 응답정보에는 Host: vhost.vdomain이라는 내용이 있음을 유의하라. 따라서 도메인 매퍼의 내부에 임의의 vhost.vdomain에 대해 임의의 host.domain으로 대응되게 하는 데이터베이스를 두면 이 도메인 매퍼가 다수의 매핑을 동시에 감당하게 할 수 있다.

서론에서 말한 예제에 대해 적용해보면 다음과 같다. 웹 사용자가 요청한 URL이 http://www.kim.com 이라면 브라우저는 네임서버에게 www.kim.com의

```
HTTP/1.1 200 OK
Date: Monday, 1-Jan-01 01:02:03 GMT
Server: Apache/5.0
MIME-version: 1.0
Content-type: text/html
Content-length: 145
<CR><LF>
.... 이하 생략 ...
```

그림 4. 웹 서버의 응답정보 (정상 트랜잭션)
Fig. 4 Reply from Web server (normal)

IP 주소를 조회하는데, 이 IP 주소가 도메인 매퍼를 가리키도록 하면 도메인 매퍼에게는 그림 3과 같은 요청정보가 보내어진다. 도메인 매퍼는 요청정보 내의 Host: 필드에 있는 www.kim.com 이라는 호스트명을 발견하고 자신의 데이터베이스를 탐색하여 이 URL에 해당되는 자원이 실제로는 www.company.com/~kim에 위치하고 있음을 알게된다. 따라서 도메인 매퍼는 그림 5와 같은 응답정보를 다시 브라우저에게 보내되, Location 필드에는 http://www.company.com/~kim을 넣어준다. 이 응답정보를 받은 브라우저는 응답코드 값이 301이므로 자신이 원하는 자원의 URL이 이동되어졌음을 알게되고, Location 필드에 있는 새로운 URL, 즉 http://www.company.com/~kim으로 접근을 시도한다. 네임서버는 www.company.com의 IP 주소를 알려주며, 브라우저는 이 IP 주소에 해당되는 웹 서버에 다시 그림 3과 같은 요청정보를 보낸다. 이 요청정보의 첫째 줄에는 GET /~kim HTTP/1.1이 들어있으므로 /~kim에 해당되는 경로의 자원을 발견하여 브라우저에게 보내어 주게 된다. 이런 일련의 과정은 웹 사용자에게는 전혀 투명하게 이루어진다. 본 연구에서는 새로운 가상 도메인명의 추가 및 삭제를 용이하게 하기 위해 mySQL[7]이라는 데이터베이스 시스템을 사용했다. 자세한 내용은 4장에서 설명한다.

3.2 네임 서버

```
HTTP/1.1 301 Redirection
Location: new_URL
```

그림 5. 웹 서버의 응답정보 (재전송 트랜잭션)
Fig. 5 Reply from Web server (redirection)

@	IN	SOA	kim.com.	root.kim.com. (
	123456789	; serial		
	86400		; refresh (6h)	
	7200		; retry (15min)	
	3600000		; expire (7d)	
	345600)		; minimum (12h)	
		IN	NS	kim.com.
www		IN	A	99.88.77.66
java		IN	A	99.88.77.66
hobby		IN	A	99.88.77.66

그림 6. URL 전달을 위한 네임서버의 zone 파일
Fig. 6 Zone file of NS for URL forwarding

그림 2에서 볼 수 있듯이 URL 전달 서버의 또 다른 중요한 구성요소는 네임서버이다. 네임서버는 기호로 표시된 도메인명에 해당되는 IP 주소가 무엇인지를 알려주는 데이터베이스로 볼 수 있다[3][8][9].

인터넷 상의 호스트는 그 수가 엄청나며, 또한 호스트의 추가 및 삭제가 빈번하므로 하나의 네임서버로서 인터넷 상의 모든 호스트를 서비스 하는 것은 불가능한 일이다. 따라서 각각의 개별 도메인마다 개별적인 네임서버를 두고 자신의 도메인에서 일어나는 호스트의 추가 및 삭제에 대한 데이터베이스 변경은 각 도메인 관리자에게 할당하며, 이런 도메인별 네임서버가 다시 상위의 네임서버에 의해 관리되는 계층적 구조의 분산 데이터베이스를 이루고 있다. 서론에서 말한 예제의 경우 company.com 도메인을 관리하는 기관에서 (대개는 해당 회사) www.company.com의 IP 주소를 담은 네임서버를 운영한다.

그러나 같은 예제에서 www.kim.com은 도메인명이 전혀 다르므로 company.com 도메인을 관리하는 기관과 상관없이 kim.com 도메인을 관리하는 개인 또는 서비스 업체에서 www.kim.com의 IP 주소를 담은 네임서버를 운영하게 된다. 이렇게 함으로서 비록 http://www.kim.com이라는 URL에 해당되는 내용이 실제로는 http://www.company.com/~kim 이라는 URL에 있다고 하더라도 kim이라는 사용자는 자신이 속한 기관인 company.com을 관리하는 네임서버에는 아무런 변경을 요청하지 않아도 된다.

kim.com 도메인을 관리하는 네임서버는 www.kim.com의 IP 주소를 반드시 그림 2(b)에서의 도메인 매퍼를 가리키는 값이 되도록 하여야 한다. 도메인 매퍼의 IP 주소를 99.88.77.66이라고 하면 이 네임서버의 zone 파일은 그림 6과 같이 될 것이다.

그림 6에서 볼 수 있듯이 www.kim.com, java.kim.com, hobby.kim.com 등은 모두 IP 주소 99.88.77.66을 가지며, 이 주소는 도메인 매퍼의 주소이므로 이들 호스트에 대한 HTTP 요청은 모두 도메인 매퍼에게로 보내진다. 도메인 매퍼는 HTTP 요청 내에 들어있는 Host: 정보를 보고 브라우저가 원하는 호스트명이 무엇이었는지를 알 수 있다(그림 3). 이 호스트명에 해당되는 가상 URL을 자신의 데이터베이스를 탐색하여 얻은 값, 즉 http://www.company.com/~kim, http://www.company.com/~kim/java, http://www.company.com/~kim/hobby 등의 내용을 그림 5의 Location: 필드에 실어 브라우저에게 전송하게 된다.

3.3 이메일 전달

이상에서 살펴 본 바와 같이 웹 브라우저가 요청한 URL을 전달하기 위해서는 도메인 매퍼라는 호스트 컴퓨터에 웹 서버와 유사한 기능을 갖는 프로그램이 있어야 하며, 이 프로그램은 데이터베이스와 연동되어 가상 도메인명으로부터 실제 도메인명을 찾게 된다. 일반적인 웹 서버는 그림 3과 같은 요청정보를 받아들여 그림 4와 같은 응답정보를 내지만, 이 프로그램은 그림 5와 같은 응답정보를 낸다는 점에서 차이가 있다.

이와 달리 이메일 전달은 별도의 프로그램 없이 단순히 도메인 매퍼 내의 메일 환경 설정 파일 내용만 바꾸어 줌으로서 이루어질 수 있다. 이메일 전달이란 user@www.kim.com 등의 주소로 전자우편이 올 때 이것을 다시 kim@www.company.com이라는 주소로 전달하여 주는 것을 의미한다. 이메일 전달을 위해서는 도메인 매퍼에 기본적으로 설치되어 있는 sendmail 프로그램의 별명(alias) 테이블만 편집해주면 된다.

sendmail 프로그램이 참조하는 별명 테이블을 포함하는 파일의 종류로는 /etc/mail/aliases 및 /etc/mail/virtusertable 등이 있으며[10], 이들 중 aliases는 자기 시스템 내에서의 이메일 주소 전달

목적으로 사용되고, virtusertable은 다른 시스템의 사용자를 포함하는 이메일 주소 전달 목적으로 사용된다. 본 연구의 목적에 부합하는 것은 virtusertable을 사용하는 것이다. 그림 7에 virtusertable의 예를 나타내었다.

이 예제에서는 user@www.kim.com, root@java.kim.com, web@hobby.kim.com 등으로 보낸 이메일이 모두 kim@www.company.com 으로 전달되어지도록 한 것이다. 그림 6에서 www.kim.com, java.kim.com, hobby.kim.com 등이 모두 도메인 매퍼의 IP 주소를 가리키도록 되어있으므로 이들 호스트에 대한 이메일은 모두 도메인 매퍼에게 보내지며, 도메인 매퍼 내의 sendmail 프로그램은 다시 virtusertable 내의 데이터베이스를 참조하여 최종 수신자인 kim@www.company.com 으로 재전송하게 된다. 물론 virtusertable 은 단순한 텍스트 파일이므로 데이터베이스화 하기 위해 다음 명령을 실행해 두어야 한다.

makemap hash /etc/mail/virtusertable.db < /etc/mail/virtusertable

실제로는 이메일 전달주소는 mySQL DBMS에 의해 저장되어 관리되어지며, 리눅스 운영체제의 cron에 의해 주기적으로 업데이트된다.

IV. 구 현

본 논문에서 제안한 URL 전달서버는 Pentium III 933MHz 프로세서를 갖는 PC 상에 구현되었다. 캐쉬 메모리는 256KB이며, 주 메모리는 128MB이다. 운영체제는 Linux kernel 2.2.14이며, 사용한 C 컴파일러는 glibc 2.1.2-3, libc 5-1.0-2 이다.

이 PC 서버 상에 도메인 매퍼 프로그램과 네임서버가 각각 설치되었다. 네임서버는 리눅스용의 bind-8.1.2-5을 설치하였으며, 이외에도 새로운 URL의 추가 및 삭제를 용이하게 하기 위해 DBMS를 사용하였다. 여기서는 리눅스용의 mySQL 3.22.23을 설치했다.

실험적으로 100개의 도메인명에 대해 각각 1000개의 호스트명을 지정하고 URL 전달실험을 해 본 결과 본 환경에서의 전달지연시간은 평균

user@www.kim.com	kim@www.company.com
root@java.kim.com	kim@www.company.com
web@hobby.kim.com	kim@www.company.com

그림 7. 도메인매퍼 /etc/mail/virtusertable 파일
Fig. 7 /etc/mail/virtusertable of domain mapper

8msec 이내로, 웹 페이지를 접속하는데 소요되는 전체 평균시간이 현재 평균 1초 이상임을 감안할 때 무시할만한 수준으로 나타났다.

구현과 관련된 세부사항은 다음과 같다.

4.1 도메인 매퍼

도메인 매퍼는 전형적인 인터넷 서버 프로그램의 형태를 따른다. 웹 브라우저의 입장에서는 도메인 매퍼가 웹 서버의 역할을 하는 것처럼 보여야 하므로 일반적인 웹 서버와 마찬가지로 HTTP 기본 포트인 80번 포트를 사용한다.

브라우저로부터 접속요청이 들어오면 도메인 매퍼는 자식 프로세스를 생성하여 그것으로 하여금 해당 브라우저를 위한 URL 전달업무를 맡게 하고, 자신은 즉시 다른 브라우저의 접속요청을 기다리는 대기상태로 들어간다. 이것은 다수의 웹 브라우저의 요청을 동시에 서비스하기 위함이며, 새로운 접속요청이 있을 때마다 그 요청을 처리해 줄 자식 프로세스를 계속해서 만들어준다.

보다 빠른 응답시간 특성을 얻기위해 최근의 웹 서버들은 시간이 많이 드는 프로세스 생성 대신 다중 쓰레드 방식을 사용하거나, 또는 접속요청이 들어온 후 자식 프로세스를 만드는 것이 아니라 처음부터 몇 개의 자식 프로세스를 생성해두는 방법 등을 사용한다. 그러나 도메인 매퍼의 경우 일반적인 웹 서버와 달리 대용량의 트랜잭션을 처리하는 것은 아니므로 본 연구에서는 고전적인 형태의 인터넷 서버 프로그램 형태를 따르게 했다.

4.2 데이터베이스

URL 전달 서버는 내부적으로 논리적 URL 명과 물리적 URL 명을 기억하고 있는 데이터베이스를 갖고 있어야 한다. 이 데이터베이스는 간단한 ASCII 파일이 될 수도 있고, 메일서버 등이 사용하는 alias 파일과 같은 이진파일이 될 수도 있

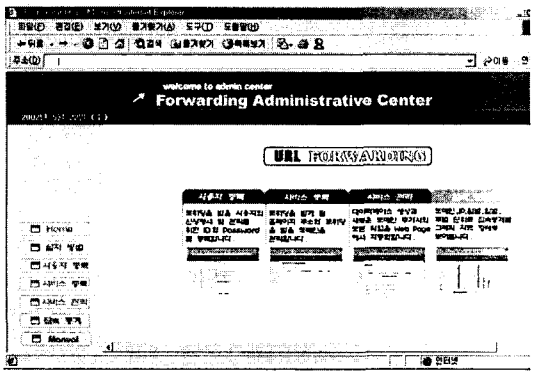

그림 8. 웹을 통한 URL 전달서버의 접속
Fig. 8 Accessing URL forwarding server thru Web

다. 데이터베이스에 들어가는 URL들의 개수가 적거나 또는 URL들에 대한 추가 및 삭제가 거의 일어나지 않는다면 그때그때마다 관리자가 이 파일을 편집기를 사용하여 직접 편집하면 되므로 이런 방법들을 쓰는 것도 무난하다. 그러나 URL의 개수가 수천개 또는 그 이상으로 많아지거나 또는 이들에 대한 추가 및 삭제가 빈번히 일어난다면 관리자가 해당 파일을 편집하는 방식으로는 감당하기 어렵게 된다.

이런 이유로 본 연구에서는 DBMS 방식에 의한 데이터베이스를 구축하여 사용했다. DBMS 사용에 따른 추가부담도 있고, 또 데이터베이스의 양이 적을 경우 파일 방식보다 오히려 늦을 수 있다는 단점도 있지만 논리적 URL 사용이 더욱 활발해지기 위해서는 유지성(maintainability) 및 확장성(scalability)이 더 중요하기 때문이다. 본 연구에서는 리눅스 환경에서 널리 쓰이는 DBMS 중 한가지인 mySQL을 사용하였다.

이 DBMS 에는 각종 도메인명에 대한 정보를 담은 테이블과, 개별 도메인명에 대한 전달정보를 담은 테이블 등이 놓인다. 후자의 테이블은 각 개별 도메인마다 별도의 테이블을 할당했으며, 도메인명이 그 테이블의 이름이 되도록 했다. 도메인명에 대한 정보를 담은 테이블의 조작은 DBMS의 관리자만이 할 수 있도록 했으며, 개별 도메인명에 대한 전달정보를 담은 테이블의 내용은 그 개별 도메인의 소유자 또는 그의 허가를 받은 임의의 사용자가 웹 브라우저를 사용하여 DBMS와 연동된 CGI 방식을 통해 온라인으로 변경할 수 있게 하였다.

4.3 네임서버 및 최신상태의 유지

DBMS 에 저장된 정보들은 다시 네임서버의 데이터베이스에 반영되어진다. 예를 들어 kim.com 도메인의 사용자가 www.company.com/~kim/family라는 물리적 URL을 family.kim.com이라는 논리적 URL로 접근할 수 있게 하려면 DBMS의 kim.com 테이블 내에 family라는 항목이 들어가야하며, 동시에 네임서버 내에 family라는 새로운 호스트명이 kim.com 도메인의 데이터베이스에 추가되어야 한다.

DBMS의 내용은 언제라도 추가, 삭제, 또는 변경할 수 있지만 네임서버와 관련된 부분은 그렇지 못하다. 네임서버는 전세계적 규모의 분산 데이터베이스를 이루고 있으며, 각 도메인명에 대한 네임서버의 내용변경은 즉시 그 효과를 나타내는 것이 아니라 미리 정해진 리프레쉬(refresh) 시간이 경과하여야만 전체 데이터베이스에 대한 정보변경이 이루어진다. 본 구현에서는 리눅스의 cron 기능을 이용하여 매 30분마다 한 번씩 DBMS 의 변경 내용을 추출하여 그것을 네임서버의 데이터베이스에 반영시키도록 하였다.

4.4 사용자 인터페이스

본 연구에서 구현된 URL 전달 서버는 웹 환경에서 환경설정, 서비스 관리, 통계 분석 등이 이루어지도록 했다. 그림 8은 URL 전달 서버에 웹 브라우저를 사용하여 접속했을 때의 첫 화면을 보여준다. URL 전달 서비스를 받기 위해서 일반 사용자들은 먼저 사용자 등록을 하여야 하며, 등록 이후 실제 전달 서비스를 위한 서비스 등록을 하게 된다. 서버의 관리자는 서비스 관리 메뉴를 통해 데이터베이스 생성, 새로운 도메인명의 추가 등을 할 수 있다. 아울러 접속통계에서는 도메인별, IP 주소별, 월별, 일별, 주별 단위의 전달 통계를 그래픽 형태로 보여줌으로서 서버의 활용도를 평가할 수 있도록 했다.

V. 관련 연구 및 비교

HTTP 프로토콜에서 재전송 트랜잭션을 포함한 것은 어떤 URL에 해당되는 인터넷 자원이 다른 위치로 옮겨졌을 때 웹 브라우저의 사용자가 별도로 다른 링크를 클릭하지 않더라도 브라우저가 자동적으로 옮겨진 위치를 찾아갈 수 있도록 하는 목적이었다. 인터넷의 특성상 파일을 비롯한 여러 자원들이 어떤 위치에서 다른 위치로 옮겨지는 경우가 많기 때문에 이런 기능이 없으면 웹 사용자들이 이전에 방문하여 이용했던 자원의 URL을 기억하고 있더라도 그것이 죽은 링크(dead link)가 되어지기 때문이다. 이런 목적으로 HTTP 프로토콜에 포함된 재전송 트랜잭션을 별도의 도메인 매퍼에 적용하여 물리적 URL 대신 논리적 URL을 사용할 수 있도록 한 것이 본 연구의 주요 아이디어라고 할 수 있다.

또한 HTTP 요청정보에 Host: 필드가 포함된 것은 하나의 서버 컴퓨터로 하여금 다수개의 웹 서버 역할을 하게 하는 가상 호스팅(virtual hosting)을 가능하게 하기 위한 것이었다. Apache 서버를 비롯한 거의 대부분의 웹 서버 프로그램이 Host: 필드를 사용하여 가상 호스팅 기능을 제공한다[11]. 그러나 이것은 URL 전달 목적이 아니라 단지 하나의 컴퓨터로서 다수의 웹 서버 역할을 시킨다는 점에서 본 연구와 구별된다. 본 연구의 URL 전달서버는 웹 서버가 놓인 컴퓨터와는 별도로 독립된 도메인 매퍼가 Host: 필드를 활용하고 있다.

국내의 경우 그동안 주요 ISP 업체에서 자사의 서버 상에 개인 홈페이지를 두는 사용자들을 대상으로 개인 도메인명인 pe.kr에 대해 URL 전달 서비스를 제공해왔다. 그러나 이때의 URL 전달 서비스는 자사의 서버 상에 놓인 개인 홈페이지에 대한 것이므로 앞에서 말한 가상 호스팅과 차이가 없다고 볼 수 있다.

최근에 들어서는 국내외적으로 URL 전달 서비스를 제공하는 업체들이 속속 생겨나고 있다. 예를 들어 welcome.to라는 도메인명을 갖는 업체에서는 http://welcome.to/korea라는 URL로 접속하면 자동적으로 물리적 URL인 www.company.com/~kim/korea 등으로 전달되어지도록 하고 있다. 그러나 이들 업체들이 채택한 URL 전달 방법의 내부구조에 대해서는 알려져 있지 않으며, URL 전달과 관련된 논문도 아직까지 출판된 것을 발견하지 못하고 있다.

본 논문은 URL 전달에 대한 이론적 근거를 다룬 처음 시도라는 점에서 의미가 있으며, 우리는 이 논문에서 도메인 매퍼와 네임서버와의 상호작용에 의해 URL 전달이 이루어질 수 있음을 보였다. 아울러 DBMS를 사용하고, 또 실제 구현을 통해 그것의 적합성을 밝혔다는 점에서 중요성이 있다. 현재는 HTTP 프로토콜에서 제시된 재전송 트랜잭션 및 요청정보의 Host: 필드를 사용하여 URL 전달을 시키고 있지만, 추후 전혀 다른 방식으로 URL 전달을 할 수 있는 새로운 방법도 모색할 수 있을 것이다. 본 연구를 바탕으로 URL 전달의 활용폭을 넓힐 수 있는 방법, 전달 효율을 더욱 향상시킬 수 있는 방법, 전달 통계자료의 활용방안 등에 대한 향후 연구가 이루어질 수 있을 것으로 기대된다.

VI. 결 론

본 논문에서는 물리적 URL 대신 논리적 URL을 사용할 수 있게 해주는 URL 전달서버의 구현에 대해 알아보았다. 물리적 URL은 특정 자원이 인터넷 상의 어느 위치에 놓여져 있는가를 알려주는 위치 지정자로서의 성격이 강하며, 따라서 그 자원의 내용 또는 특색을 나타내는 이름으로는 부적절한 면이 있었다. 또한 웹의 이용이 활발해짐에 따라 소속기관의 웹 서버에 자신의 홈페이지를 두는 개인들도 많아지게 되었는데, 이런 홈페이지들에 대한 물리적 URL은 그 길이가 길어 기억하기 어려울 뿐 아니라 홈페이지의 성격을 나타내는 용도로는 어울리지 않는 이름을 갖는 경우가 많다. 논리적 URL은 자원의 실제 위치보다는 그 자원의 성격을 잘 들어낼 수 있는 임의의 이름을 부여할 수 있게 한다는 장점을 제공한다. 본 연구의 URL 전달서버는 웹 주소 및 이메일 주소에 대한 전달을 가능하게 하고 있다. URL 전달서버는 도

메인 매퍼와 네임서버의 상호작용에 의해 이루어지며, 본 논문에서는 이 상호작용에 대한 상세 고찰과 함께 실제 구현에 대해 알아보았다. URL 전달에 따른 오버헤드에 대해 이론적 분석을 해보았으며, 실험 결과 그것에 따른 시간지연은 8msec 정도로, 현재 인터넷 상에서 웹 페이지를 접근하는데 소요되는 평균 시간에 비해 무시할만한 수준임을 알 수 있었다. 향후 연구방향으로는 논리적 URL의 또 다른 활용, 전달 효율의 향상을 위한 다른 전달방법의 개발, 전달 통계자료의 활용 등이 있다.

참고문헌

[1] P. V. Mockapetris, "Domain Names - Concepts and Facilities", *RFC1034*, Nov 1987

[2] P. V. Mockapetris, "Domain Names - Implementation and Specification," *RFC1035*, Nov 1987

[3] P. Albitz and C. Liu, *DNS and Bind*, 3rd ed., O'Reilly, 1998

[4] T. Berners-Lee, L. Masinter, and M. McCahill, "Uniform Resource Locators (URL)," *RFC 1738*, December 1994

[5] T. Berners-Lee, R. Fielding, and H. Frystyk, "Hypertext Transfer Protocol -- HTTP/1.0," *RFC 1945*, May 1996

[6] J. Gettys, et al, "Hypertext Transfer Protocol -- HTTP/1.1," *RFC2616*, June 1999

[7] R. J. Yarger, G. Reese, and T. King, *MySQL & mSQL*, O'Reilly, 1999

[8] C. S. Giap, Y. Kadobayashi, and S. Yamaguchi, "Zero Internet Administration Approach: The Case of DNS," *Proc of the 12th International Conf. on Information Networking (ICOIN)*, pp.350-355, 1998

[9] P. Huck, M. Butler, A. Gupta, and M. Feng, "A Self-Configuring and Self-Administering Name System with Dynamic Address Assignment," *ACM Trans. on Internet Technology*, vol.2, no.1, pp.14-46, Feb 2002

[10] C. Hunt, *Linux Sendmail Administration*, Sybex, 2001

[11] C. Aulds, *Linux Apache Web Server Administration*, Sybex, 2000

저자소개

양희재(Heejae Yang)

1985년 부산대학교 자공학과(공학사)

1987년 KAIST 전기전자공학과(공학석사)

1991년 KAIST 전기전자공학과(공학박사)

1991년-현재 경성대학교 컴퓨터공학과 교수

※관심분야: 컴퓨터구조, 내장형시스템, 인터넷기술