韓國 컴퓨터情報學會 論文誌
第13卷 第5號, 2008. 9.

2008-13-5-4-5

영속적 링크를 제공하는 XRI 기반의 웹 링크 구조

정의현*, 김 원**, 박찬기***

A Web Link Architecture Based on XRI Providing Persistent Link

Eui-Hyun Jung*, Weon Kim**, Chan-Ki Park***

요 약

차세대 웹은 표현 중심의 웹을 데이터 중심의 웹으로 이끌 것으로 예측되며, Web 2.0과 시맨틱 웹의 기술적 결합이 될 것이다. 차세대 웹은 시맨틱 처리, 웹 플랫폼과 데이터 결합이 매우 중요한 기술적 요소이다. 이를 가능하게 하기 위해서, 필수적인 기술 중의 하나가 링크 깨짐(Link Rot) 현상을 해결하는 것이다. 링크 깨짐 현상은 단순히 사용자의 불편함을 야기하는 것에 그치지 아니하고, 정보 시스템 분야에서 데이터 무결성, 정보 손실, 서비스 결손 등의 다양한 문제들의 원인이 되고 있다. 본 논문에서는 웹의 고질적인 문제점인 링크 깨짐 현상을 해결하기 위한 웹 링크 구조에 대해 제안한다. 제안된 웹 링크 구조는 OASIS에서 제안한 XRI 기반으로 구성되었으며, URL rewriting 기능과 결합하여 영속적 링크 (persistent link)를 지원하도록 설계하였다. 서버-사이드(server-side) 기술로 설계되었기 때문에 기존의 방식에 비해 상호운용성, 투명성, 적용성 면에서 우수하다. 이에 더하여, 해당 구조는 상황인지 링크 변환에 사용될 수 있는 메타데이터 식별 기능을 제공한다.

Abstract

Web 2.0 and Semantic Web technology will be merged to be a next generation Web that leads presentation-oriented Web to data-centric Web. In the next generation Web, semantic processing, Web platform, and data fusion are most important technology factors. Resolving the Link Rot is the one of the essential technologies to enable these features. The Link Rot causes not only simple annoyances to users but also more serious problems including data integrity, loss of knowledge, breach of service, and so forth. We have suggested a new XRI-based persistent Web link architecture to cure the Link Rot that has been considered as a deep-seated problem of the Web. The proposed architecture is based on the XRI suggested by OASIS and it is designed to support a persistent link by using URL rewriting. Since the architecture is designed as a server-side technology, it is superior to existing research especially in Interoperability, Transparency and Adoptability. In addition to this, the architecture provides a metadata identification to be used for context-aware link resolution.

▸ Keyword : Link Rot, Persistent Link, URL Rewriting, URN

• 제1저자 : 정의현
• 접수일 : 2008. 6. 29, 심사일 : 2008. 8. 11, 심사완료일 : 2008. 9. 25.
* 안양대학교 컴퓨터학과 전임강사 ** 인터넷 진흥원 기술개발단 단장 ***인터넷 진흥원 기술개발단 팀장
※ 이 연구는 2008년도 한국 인터넷 진흥원 공동 연구과제의 결과물임

Ⅰ. 서 론

웹이 대두된 이후로 웹의 링크 깨짐(Link Rot) 현상은 해결되기 어려운 문제로 간주되어 왔다[1]. 따라서 일반 사용자가 웹 서핑(surfing) 시에 404 HTTP 에러 (Page Not Found)를 만나는 것은 흔한 문제이며, 특별한 주목의 대상이 되지 못했다. 그러나 링크 깨짐 현상은 단순히 사용자의 불편함을 야기하는 것에 그치지 아니하고, 정보 시스템 분야에서 데이터 무결성 (Data Integrity), 정보 손실, 서비스 결손 등의 다양한 문제들의 원인이 되고 있다. 이러한 이유 때문에 링크 깨짐은 웹의 창시자인 팀 버너스리를 포함한 여러 문헌에서 그 문제점이 자세히 언급되고 있다[1,2,3,5].

문제가 더욱 복잡한 것은 URL (Unified Resource Locator)이 여러 분야에서 다양하게 사용됨에 따라, 링크 깨짐 현상이 온라인 웹 서비스에만 국한되지 않는다는 점이다. 예를 들어, 과학 문헌에서 웹에서만 찾을 수 있는 정보를 참조(reference)하는 것은 일반적인 현상으로 정착되고 있으며, 그 빈도도 점차 늘어나고 있다. 그러나 과학 문헌에서의 이러한 URL 참조는 시간이 지날수록 무결성에 문제가 발생할 수밖에 없는 한계점을 안고 있다. Sellitto[3]의 연구에 의하면 1995년부터 2003년 사이에 발간된 123개의 학술회의의 논문을 조사한 결과, 1,068개의 웹 참조 중 22%의 참조 링크가 깨진 것을 발견하였다. 즉, 과학 문헌에서의 URL 참조는 온라인 웹에서와 마찬가지로 데이터의 무결성 문제와 정보 손실을 필연적으로 초래하게 된다 [4]. 또한 인터넷 자료 관련 전문가들은 웹 페이지의 평균 수명이 100일 정도라는 것을 밝혀내었다 [5].

이러한 문제점을 해결하기 위하여 다양한 연구들이 수행되었다[6,7,9]. 일부 연구자들은 링크 영속성(link consistency)을 향상시키기 위해서는 기존 웹 프로토콜이 수정되어야 한다고 주장하였다[7,8]. 이 중, W3Objects[7]는 객체지향 방식을 적용하여 웹 리소스를 리소스와 내부 상태, 기능을 갖는 객체로 표현할 것을 제안하였다. 또한, Hyper-G[8]의 경우에는 기존 웹을 새로운 링크 영속성을 제공하는 새로운 웹 구조를 갖자는 좀 더 급진적인 방식을 제안하였다. 이러한 접근 방법이 그 자체로 흥미로운 방식이기는 하지만, 기존 웹과의 상호운용성면에서 실패하였기 때문에 널리 보급되지는 못하였다.

다른 접근 방안으로는 URN (Uniform Resource Name) 규약[9]이 있다. URN은 웹 리소스에 불변의 식별자를 주고자 하는 목표로 설계되었다. 그러나 URN의 명확한 해결방안에도 불구하고, 웹 분야에서는 널리 받아들여지지 않았다. 이것은 URN이 DNS (Domain Name System)과 마찬가지로 변환(resolution) 시스템을 요구하기 때문이며, 현재의 웹 브라우저에서는 이에 대한 지원을 전혀 하지 않기 때문이다. 이러한 이유 때문에, URN 기술의 보급과 개발을 위하여 PURL (Persistent URL)[10] 연구가 제안되었다. PURL은 웹 리소스의 논리적 위치를 명시하고, 해당 논리 위치에 대한 요청이 들어오는 경우에 HTTP 리다이렉션 (redirection) 기능을 이용하여 실제적 위치로 매핑 시켜주는 구조를 제공한다. 이러한 간접적인 위치 지정 방식은 기존 웹과의 호환성 측면에서 더 향상된 링크 영속성(link persistency)을 제공할 수 있었다. 그러나 이러한 가상 URL 방식은 추가적인 네트워크 트래픽을 야기한다는 문제점과 북마크를 제대로 지원하지 못하는 몇 가지 문제점을 노출하였다. 이 중에서 가장 심각한 문제점은 PURL도 URN과 마찬가지로 별도의 PURL 변환 서버가 요구된다는 점이다. DNS 서버와 마찬가지로 PURL 변환 서버도 가상 URL에 대한 변환 레코드를 PURL 변환 서버에 관리자가 개입하여 수작업으로 관리해주어야만 한다. 이러한 이유 때문에 자신의 고유한 영속적 링크를 동적으로 생성해서 사용하고자 하는 웹 애플리케이션들에게 PURL 기술이 같이 사용되기는 어려웠다.

그 외의 접근 방안으로는 복제(replication) 방식이 있다. Internet Archive[11]는 대상(target) 웹 콘텐츠를 복제하여 저장하고, 해당 저장에 대한 영속적 링크를 제공한다. Web Cite[12]도 대상 웹 페이지를 복제한 후에 참고 문헌에서 사용 가능한 형태의 영속적 링크를 제공한다. 이러한 접근 방안은 특정한 웹 페이지들에 대해서 영속적 링크를 제공하는 유용한 방안이 될 수 있지만, 제공되는 모든 영속적 링크가 서비스를 제공하는 웹 사이트에 한정된다는 단점을 갖는다. 당연히 이것은 확장성(scalability)에 문제를 야기하게 되며, 또한 이 방식은 외부 웹 애플리케이션과 통합할 수 있는 방법이 전혀 없다는 한계점을 갖고 있다. 지금까지 연구된 다양한 접근 방안은 일부 측면에서 링크 영속성을 향상시켰으나, 그 어떤 방안도 아직까지 널리 인정받거나 사용되지 못하고 있다.

본 논문에서는 기존 연구들을 주의 깊게 분석한 후에, 영속적인 웹 링크를 설계하는데 있어서 2가지 요소를 고려해야 한다는 결론을 얻었다. 첫째, 링크 영속성을 구현하기 위하여 URL을 사용해서는 안 된다는 것이다. 기본적으로 URL은 리소스의 위치를 나타내기 위한 것이 주요한 목적이다. 따라서 웹 상에서의 리소스 이동이 피할 수 없는 현실이기 때문에, 리소스의 위치를 나타내는 URL로 링크 영속성을 제공하려는 어떤 시도도 실패할 수밖에 없다는 사실을 이해해야 한다. 두 번째로, 아무리 우수한 영속적 링크 구조를 제안한다 하여도, 그

영속적 링크 구조가 널리 사용되기 위해서는 반드시 기존 웹과의 상호운용성(Interoperability), 사용자 입장에서의 투명성(Transparency), 외부 웹 애플리케이션 입장에서의 적용성(Adoptability) 등을 우선적으로 고려해야 한다는 점이다.

본 논문에서는 이러한 결론을 바탕으로 새로운 영속적 웹 링크 구조인 XRILink를 제안한다. XRILink는 XRI(Extensible Resource Identifier) [13] 기술과 URL Rewriting[14] 기술에 근거하고 있다. XRI는 OASIS에서 개발된 새로운 추상적 식별체계 기술이다. XRI는 웹과 정보통신 분야에서 영속성(persistency)과 메타 데이터(metadata) 기능을 가진 식별체계를 제공할 수 있는 유력한 기술로 간주되고 있다. 그러나 XRI가 웹에서 사용될 때는 기존 연구들이 경험했던 것과 똑같은 상호운용성의 약점을 갖고 있다. 비록 XRI가 URI에 기반을 두고 있지만, 영속적 링크로 사용되는 경우의 XRI 자체는 현재의 웹 브라우저에서 지원되지 않고 있는데, 이는 영속적 XRI 링크는 URN과 마찬가지의 변환 서비스를 필요로 하기 때문이다. 본 논문에서는 이러한 문제를 해결하기 위하여 URL Rewriting 기술을 제공하는 Rewrite Engine[14]을 사용하여 이 문제를 해결하였다. XRI와 URL Rewriting 기술을 결합함으로써, XRI는 URL과 마찬가지의 일반적인 하이퍼링크로 사용될 수 있으며 동시에 링크 영속성을 제공할 수 있게 된다.

본 논문의 구성은 2장에서 설계 고려 사항에 대해서 살펴보고, 3장에서 시스템의 구조에 대해서 논한다. 그리고 4장에서 실제 구현 및 평가 사항에 대해서 살펴본다. 마지막으로 5장에서 결론과 향후 과제 방향에 관해서 논한다.

II. 설계 고려 사항

URL과 달리 URN은 인터넷 리소스를 영속적이고, 리소스의 현재 위치에 독립적인 식별자를 제공하기 위해 개발되었다. 그러나 설계 목적의 우수함에도 불구하고, URN을 위한 변환 서비스는 실제로는 널리 사용되지 않았다. 물론 웹 브라우저에는 "urn" 속성이 HTML <a> 태그에 아직까지 제공되어 있다. 예를 들어, 특정 저널에 대한 URN인 "urn:issn:0167-6423"을 HTML에 다음과 같이 입력할 수 있다.

```
<a urn="urn:issn:0167-6423">Science of Computer</a>
```

웹 브라우저에서는 이렇게 URN이 포함된 링크에 대해서도 하이퍼링크로 표시는 해주지만, 실제 하이퍼링킹 동작이 일어나지는 않는다. 즉, URL은 리소스의 실제 위치를 나타나는 데는 장점이 있지만, 영속적 링크를 나타내는 데는 성공적이지 않으며, URN은 URL과 반대의 특성을 지니고 있다.

XRI는 추상적 식별자(abstract identifier)를 위한 식별체계 구조와 변환 프로토콜의 집합이다. 추상적 식별자란 대상 리소스를 지칭하는 영속적 식별자로서 실제 사용 시에는 리소스의 주소나 속성으로 변환되게 된다. XRI는 영속적이고 메타데이터의 지원이 가능한 확장 식별체계로서 지능적이고 확장이 용이한 구조를 갖고 있다 [15]. XRI의 기능 중에서 교차 참조(Cross Reference)[13]는 여러 식별자를 합쳐서 새로운 식별자를 만들 수 있는 매우 유용한 기능이다. 이것을 이용함으로써, 특정 저널의 URN 식별자인 "urn:issn:0167-6423"를 <표 1>과 같이 도서관이나 서점의 URL과 결합시킬 수 있다.

표 1. XRI에서의 교차 참조의 예
Table 1. Examples of Cross Reference in XRI

xri://book_store.foo.com/(urn:issn:0167-6423)
xri://library.com/(urn:issn:0167-6423)

이 기능은 하나의 식별자로 위치와 영속적인 식별을 동시에 가능하게 해주는 특징을 갖고 있다. 이것은 URN과 URL의 장점만을 합친 듯이 사용될 수 있으며, 일반적인 URI 처럼 웹 브라우저에 삽입될 수 있는 특징을 갖고 있다. 이것은 또한 전역적(global)인 전담(dedicated) 변환 서버의 도입 대신에 애플리케이션들마다 특정 변환 서버를 지정할 수 있는 가능성을 보여준다. 그러나 이 기능만으로 기존 웹과 상호운용성이 높은 영속적 링크를 제공하는 데는 부족한 면이 있다. 교차 참조의 XRI에서 변환 서버의 URL 파트는 브라우저에게 대상 서버의 IP 주소만을 알려주는 역할만을 한다. 이것이 의미하는 바는 해당 링크 안에 변환 서비스를 담당할 CGI 핸들러(handler)가 반드시 명시되어야 한다는 점이다.

예를 들어, <표 1>의 링크들이 대상 서버에서 제대로 변환되어 사용하기 위해서는 HTTP XRI(HXRI)[16]과 CGI 핸들러가 추가된 다음의 형태로 수정되어야 한다. 그러나 이렇게 수정된 식별자는 더 이상 영속적 식별자가 아니며, 영속적 식별 링크를 변환하기 위해 요청되는 단순한 CGI URL일 뿐이다. 사용자 입장에서는 링크 영속성과 관련 없는 무의미한 핸들러 이름을 별도로 외우는 것은 문제가 있으며, 영속적 링크의 유지 기간도 짧아질 수밖에 없다. 그러나 이 수정이 링크 영속성을 심하게 훼손함에도 불구하고, CGI 핸들러를 명시하는 것은 XRI가 현재의 웹과 결합하기 위해서는 피할 수 없는 문제이다.

http://library.com/handler?urn=(urn:issn:0167-6423)

영속적 링크는 문자 그대로 영원불변(permanent)의 링크를 의미하는 것은 아니다. 왜냐하면 해당 사이트가 사라지거나 옮기면 해당 링크가 사라지기 때문이다. 그러나 한번 외부로 노출된 영속적 링크는 해당 사이트나 해당 애플리케이션이 유지되는 동안에는 영속적이어야 한다. 그러나 CGI 핸들러의 명시는 새로운 서버 사이드(server-side)의 기술이나 사이트 구조의 변경, 개발자의 교체 등으로 예측되지 않는 방향으로 변경될 수 있다. 당연하게도 이러한 변화는 해당 서버의 모든 영속적 링크를 일거에 쓸모없게 만들 수 있다. 즉, CGI 핸들러의 명시가 XRI를 현재의 웹과 연동하는데 필수적이지만, 결정적으로 XRI가 가장 장점으로 내세우는 링크 영속성을 망치는 결과를 갖게 한다.

아직까지는 이러한 모순을 해결할 수 있는 방안을 XRI 연구진들은 내놓지 않고 있다. 의심할 바도 없이, XRI는 영속적 링크에 대한 유용한 방안이지만, 현재의 웹 구조와 적절히 결합할 방법을 찾지 못한다면, URN과 마찬가지의 길을 가게 될 것이다. 이러한 모순을 해결하기 위하여 본 논문에서는 Rewrite Engine[14]를 사용하는 방법을 제안한다. Rewrite Engine은 대부분의 웹 서버에서 제공하는 방식으로서 웹 서버로 들어오는 URL을 해당 콘텐츠와 매칭하기 이전에 URL의 내용을 수정하는 모듈이다. 본 논문에서는 이를 이용하여 웹 서버로 들어오는 영속적 링크를 외부의 인지 없이 내부적으로 특별한 핸들러를 추가시키도록 구성하였다. 이것이 의미하는 바는 URL Rewriting을 적절히 이용하면, 노출된 영속적 링크가 해당 사이트의 내부적 변화와 무관하게 영속성을 유지할 수 있다는 점이다. 또한, 이러한 서버 사이드 접근 방안은 메타 데이터 식별 기능[15]의 장점도 제공할 수 있다. 메타 데이터 식별 기능은 상황인지(context-aware)나 QoS 기반의 링크 변환을 제공할 수 있는 식별 기능인데, XRI 식별체계에서는 단순히 URL 만을 제공하는 것이 아니고, 부가적인 메타 데이터를 제공할 수 있기 때문이다.

III. 시스템 구조

제안된 시스템의 구조는 〈그림 1〉에서 보는 바와 같이 크게 Rewrite Engine, 변환 핸들러, 변환 서버의 3가지 컴포넌트로 구성된다. 기본 구조가 서버 사이드 기술에 근거하고 있기 때문에 클라이언트 입장에서 새로운 모듈을 도입해야할 필요가 없다. 이러한 구조는 웹 브라우저의 수정 없이도 XRI를 보통의 URL처럼 쓸 수 있다는 것을 의미한다. 웹 브라우저에서는 XRI 링크의 서버 파트를 이용하여 XRI 링크의 영속적 파트를 웹 서버로 넘겨주게 된다. 이렇게 넘겨진 영속적 파트는 변환 핸들러에게 파라미터로 넘어가게 된다.

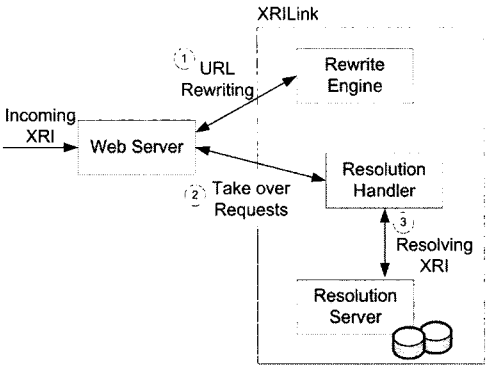

그림 1. 제안된 시스템의 컴포넌트들과 그 관계
Fig 1. Components of XRILink and their relationship

예를 들어 다음의 링크가 있는 경우에,

〈a href="http://library.com/(urn:issn:0167-6423)"〉
Science of Computer
〈/a〉

"Science of Computer" 하이퍼링크를 선택하면, 이 링크의 영속적 파트인 "urn:issn:0167-6423"가 "library.com" 서버의 Rewrite Engine에 넘겨지게 된다. 그러면 해당 엔진은 들어온 URL을 변환 핸들러의 이름이 추가된 새로운 URL로 바꾸게 된다. 예를 들어, "urn:issn:0167-6423"의 경우에는 다음과 같은 URL로 바꾸게 될 것이다.

xri.php?xri=(urn:issn:0167-6423)

이렇게 바뀐 URL은 웹 서버에 의해서 해당 변환 핸들러(여기서는 xri.php)로 URL을 넘겨주게 된다. 그러면 핸들러는 영속적 파트인 "urn:issn:0167-6423"을 변환 서버에 요청하여 XRI Resource Descriptor Sequence (XRDS) [16] 문서를 받게 된다. 이러한 변환 과정이 완결된 후에, 변환 핸들러는 XRDS 문서를 파싱하고, 적절한 콘텐츠를 웹 브

라우저에 넘겨주게 된다. 이러한 절차는 〈그림 2〉의 시퀀스 다이어그램(Sequence Diagram)에 묘사되어 있다.

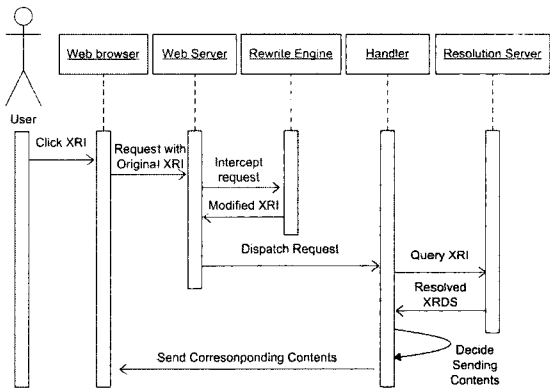

그림 2. XRI 변환의 시퀀스 다이어그램
Fig 2. Sequence Diagram of XRI resolution

IV. 구현 및 평가

기존 연구들과 달리 제안된 시스템은 특정 기술을 강제하지 않으며, Ruby, JPS, 혹은 아파치 모듈과 같은 대부분의 서버 사이드 기술에서 구현이 가능하다. 따라서 기존의 웹 애플리케이션은 물론이고, 향후 새로운 웹 애플리케이션에서도 주로 사용되는 기술과 상관없이 쉽게 적용이 가능하다. 본 논문의 프로토타입 구현에서는 mod_alias[14] 아파치 모듈이 Rewrite Engine으로 사용되었다. mod_alias 모듈은 아파치 서버에 기본으로 장착된 기본 모듈로서 서버 입장에서 새롭게 설치할 필요가 없다. 그러나 이 기본 모듈만으로도 제안된 시스템이 필요로 하는 Rewrite 기능을 충분히 제공할 수 있었다. 만일 이보다 복잡한 기능이 필요한 경우에는 mod_rewrite[14]와 같이 다양한 기능을 제공하는 모듈이나 사용자가 직접 만든 모듈을 사용할 수도 있다.

mod_alias 모듈의 rewrite 규칙을 설정하기 위하여, 다음의 스크립트를 테스트 아파치 웹 서버의 httpd.conf 파일에 추가하였다.

```
AliasMatch ^/\((.*)"/usr/local/apache2/htdocs/xri.php/($1"
```

변환 핸들러를 만드는데는 PHP가 선택되었지만, 앞에서 언급한 바와 같이 어떠한 기술을 사용해도 무방하다. PHP로 만들어진 변환 핸들러의 코드 일부분은 〈그림 3〉과 같다.

```
// get XRI part
$xri = $_SERVER['PATH_INFO'];
$xri = strstr($xri, "(");

// resolve XRI using resolution server
// xri_resolve() is a function that queries
// to resolution server
$xrd_data = xri_resolve($xri);

// Parse XRDS data from the resolution server
$xrd_parser = new XRDSParser();
$xrd_parser->parse($xrd_data);

// decide contents to be sent
// if user is in VPN,send high QoS data
// any user defined policy can be used
$user_ip = $_SERVER['REMOTE_ADDR'];
if(inVPN($user_ip)) {
    $media_type =
    $xrd_parser->getMediaType("high");
    $target_uri = $xrd_parser->getURI("high");
} else {
    $media_type =
    $xrd_parser->getMediaType("low");
    $target_uri = $xrd_parser->getURI("low");
}

// return corresponding contents to browser
$file_path =
$_SERVER['DOCUMENT_ROOT'].'/'.$target_uri;
header("Content-Length: ".@filesize($file_path));
header('Content-Type: '.$media_type);
readfile ($file_path);
```

그림 3. 변환 핸들러의 PHP 일부 코드
Fig 3. PHP code snippet of resolution handler

해당 코드는 변환 서버와 같이 XRI 링크의 영속적 파트를 처리해주는 부분이다. 또한, 이 코드는 변환 서버로부터 XRDS 문서를 읽어들여 요청한 브라우저의 IP에 따라서 다른 콘텐츠를 전송하게 된다. 만일 요청한 브라우저가 같은 조직 내에 있는 것이라면 고화질의 비디오 콘텐츠를 반환하고, 그렇지 않은 경우에는 일반적인 HTML 문서를 반환한다. 이러한 차별적 링크 변환은 XRI의 메타 데이터 식별 기능을 이용하여 만들 수 있다. 이러한 메타 데이터 식별 기능은 고도의 상황인지 링크 변환에 사용될 수 있고, 이는 다양한 응용

의 개발에 이용할 수 있다. 프로토타입 구현에서 변환 서버는 MySQL을 이용한 간단한 쿼리 모듈로 구현하였다. 구현 서버는 다양한 변환 기술을 이용할 수 있기 때문에 해당 모듈은 다른 모듈과 외부 사용자에게 영향을 미치지 않고, 국가적인 차원(national-wide)의 변환 서비스로 교체할 수 있다.

영속적 링크의 기능을 평가하는데 있어서는 여러 가지 요소들이 고려될 수 있다. 기존 웹 아키텍처와 상호운용성이 높아야 하며, 다른 웹 애플리케이션에서 쉽게 사용이 가능해야 하고, 웹 브라우저에서 의미있는 북마크 기능을 제공할 수 있어야 하며, 웹 사용자가 투명하게 사용할 수 있는 기능이 있어야 한다. 본 논문에서는 제안된 시스템을 4가지 요소를 가지고, 기존의 연구와 비교하였다. 비교 내용은 〈표 2〉와 같다. W3Object는 새로운 웹 구조를 제안했기 때문에 상호운용성과 적용성 면에서 문제점을 노출하고 있다. PURL의 경우에는 상호운용성은 우수하지만, 외부 웹 애플리케이션에서 이를 적용할 수 있는 방법이 제공되지 못하였다. WebCite의 경우에도 중복된 콘텐츠의 URL이 중앙집중적으로 처리되기 때문에 외부 웹 애플리케이션과의 연동성에 문제점을 보여주었다. 또한 PURL과 WebCite는 가상 URL이기 때문에 사용자 입장에서 자연스러운 URL 형태가 되지 못하여, 투명성과 의미 있는 북마크의 지원에 약간의 문제점을 노출하였다. 이에 비해 본 연구가 제안한 구조는 해당 평가 요소에 대해 모두 우수한 기능을 보여주었다.

표 2. 영속적 링크 제공 연구들의 비교
Table 2. Comparison of persistent link technologies

	상호운용성	투명성	적용성	북마크 지원
W3Object	X	○	X	○
PURL	○	△	X	△
WebCite	○	△	X	△
XRILink	○	○	○	○

○: 좋음, △: 보통, X: 나쁨

V. 결론 및 향후 과제

웹상에서의 영속적 링크는 데이터 무결성을 향상시키는 점뿐 아니라 웹 2.0과 같은 새로운 웹 서비스의 다양한 요구들을 만족시키기 위해서 점차적으로 그 중요성이 주목받고 있다. 그러나 영속적 링크는 오랫동안 논쟁이 많은 연구 주제였고, 아직까지도 완전한 해결책이 제시되지 않았다. 본 논문에서는 XRI와 Rewrite Engine을 결합하여, 링크 깨짐 현상을 해결할 수 있는 새로운 웹 링크 구조를 제안하였다. 제안된 구조는 기존 연구들에 비해서 상호운용성, 투명성, 적용성, 북마크 지원 등이 우수하였다. 제안된 구조는 링크 영속성이 필수적인 Blog 애플리케이션이나 과학문헌 참조 등의 다양한 정보 시스템에 쉽게 적용이 가능한 구조를 갖고 있다. 본 연구의 초기 작업은 인터넷 진흥원과 공동으로 연구가 진행되고 있으며, 향후 국내의 차세대 웹 링크 구조 연구에서 활용될 계획이다.

참고문헌

[1] D. Spinellis, "The Decay and Failures of Web References," Communications of ACM, vol. 46, no.1, pp.71-77, Jan. 2003.

[2] T. Berners-Lee, "Cool URIs don't change," W3C style guide, "http://www.w3.org/Provider/Style/URI", 1998.

[3] C.Sellitto, "The impact of impermanent Web-located citations: A study of 123 scholarly conference publications," Journal of the American Society for Information Science and Technology, vol. 56, no. 7, pp. 695-703, Mar. 2005

[4] 이우기, "하이퍼텍스트 정보 관점에서 의도적으로 왜곡된 웹 페이지의 검출에 관한 연구," 컴퓨터정보학회 논문집 10권 1호, 2005

[5] K. Mayfield, "Wayback Goes Way Back on Web, "http://www.wired.com/culture/lifestyle/news/2001/10/47894", Wired, 2001.

[6] 정의현, 김원, 송관호, 박찬기, "웹 2.0을 위한 다국어 식별자 기반의 Cool URI에 대한 연구," 컴퓨터정보학회 논문집 11권 5호, 2006

[7] D. B. Ingham, S. J. Caughey, and M. C. Little, "Fixing the "Broken-link"Problem: The W3Objects Approach," Computer Networks & ISDN Systems, vol. 28, no.7-11, pp.1255-1268, May, 1996.

[8] F. Kappe, K. Andrews, and H. Maurer, "The Hyper-G Network Information System," Journal of Universal Computer Science, vol. 1, no.4, pp.206-220, 1995.

[9] K. Sollins and L. Masinter, "Functional Requirements for Uniform Resource Names," IETF RFC 1737, Dec. 1994.
[10] K. Shafer, S. Weibel, E. Jul, and J. Fausey, "Introduction to persistent uniform resource locators," OCLC Computer Library Center, Inc., "http://purl.oclc.org/docs/inet96.html".
[11] Internet Archive,"http://www.archive.org".
[12] G. Eysenbachand M. Trudel, "Going, Going, Still There: Using the WebCite Service to Permanently Archive Cited Web Pages,"Journal of Medical Internet Research, vo.7, no.5, e60, Oct.-Dec. 2005.
[13] D. Reed, D. McAlpin, P. Davis, N. Sakimura, M. Lindelsee, G. Wachob, "Extensible Resource Identifier (XRI) Syntax V2.0," OASIS Committee Specification, Nov. 2005
[14] R.S. Engelschall, "URL Rewriting Guide," Apache HTTP Server Official Document, "http://httpd.apache.org/docs/2.2/en/misc/rewriteguide.html", Dec. 1997.
[15] D. Reed, D. McAlpin, F. Labalme, M. Lindelsee, and G. Wachob, "An Introduction to XRIs," OASIS Working Draft 04, Mar. 2005.
[16] G. Wachob, D. Reed, L. Chasen, W. Tan, S. Churchill, D. McAlpin, C. Sabnis, P. Davis, V. Grey, M. Lindelsee, "Extensible Resource Identifier (XRI) Resolution V2.0," OASIS Working Draft 10, Mar. 2006.

저 자 소 개

정 의 현

1992년 한양대학교 전자공학사
1994년 한양대학교 전자공학석사
1999년 한양대학교 전자공학박사
1999년 ~ 2002년 대우통신 선임연구원
2004년 ~ 현재 : 안양대학교 컴퓨터학과 전임강사
관심분야 : 시맨틱 웹, 디지털 컨버전스, 센서 네트워크

김 원

1984년 한양대학교 전자공학사
1989년 한양대학교 전자공학석사
2002년 경희대학교 전자공학박사
1984년 ~ 1987년 국방과학연구소 연구원
1989년 ~ 1992년 데이콤 대리
1989년 ~ 1999년 한국전산원 선임연구원
1999년 ~ 현재 : 한국인터넷 진흥원 단장

박 찬 기

1994년 美 George Mason대학교 전산학 학사
1994년 ~ 1995년 선경정보시스템 사원
1995년 ~ 1999년 한국전산원 주임연구원
1999년 ~ 현재 한국인터넷진흥원 기술연구팀 팀장